BACHELOR THESIS

IN DEM STUDIENGANG WIRTSCHAFTSINFORMATIK PLUS

Titel:	Evaluation des In-Memory Datenbanksystems SAP HANA als Cloudlösung für Business Intelligence Systeme
Erstellt von:	Benjamin Wieder Königswaldstraße 3 77656 Offenburg
Matrikelnummer:	169479
Bearbeitungszeitraum:	01.09.2012 – 19.02.2013
Betreuer:	Prof. Dr. Stephan Trahasch
Zweitbetreuer:	Prof. Dr. Jan Münchenberg

Kurzfassung

Global agierende Unternehmen sind mit einem stetig ansteigenden Datenaufkommen konfrontiert. Zur Analyse werden diese Daten in Business Intelligence Systeme geladen. Aufbereitete Daten werden heutzutage in der Regel in zeilenbasierten (relationalen) Datenbanksystemen vorgehalten. Ein großer Nachteil dieses Datenbanksystems für diesen Anwendungsfall ist, dass es auf eine hohe Anzahl einfacher Transaktionen mit wenigen Datensätzen ausgelegt ist. Im Kontext eines analysebasierten Modells, wie hier beim Business Intelligence, fallen dagegen hauptsächlich komplexe Abfragen auf einer Vielzahl von Datensätze an [BaGü2009].

Die SAP AG stellt mit dem Produkt SAP HANA eine Lösung zur Verfügung, welche auf die Datenanalyse für Business Intelligence ausgelegt ist und zugleich auch transaktionalen Anforderungen genügen soll. Das System ist als Appliance aus Soft- und Hardware konzipiert, das auf einem spaltenorientierten, In-Memory Datenbanksystem aufbaut.

Diese Arbeit geht zunächst auf die Grundlagen des Business Intelligence, die allgemeinen Prinzipien von spaltenorientieren In-Memory Datenbankensystemen sowie deren Abgrenzung zu klassischen relationalen Systemen ein. Weiterhin wird auf die Architektur von SAP HANA und deren Besonderheiten eingegangen. Hauptgegenstand dieser Arbeit ist die Umsetzung einer Demoanwendung basierend auf SAP HANA in einer Cloud Umgebung (SAP HANA ONE). Hierbei wird insbesondere auf die Realisierbarkeit der Anwendung mittels der Entwicklungsumgebung SAP HANA Studio in einer Cloudumgebung eingegangen. Weiterhin steht zur Diskussion inwiefern SAP HANA in bestehende Business Intelligence Umgebungen integriert werden kann.

Vorwort

Mein Dank gilt Prof. Dr. Stephan Trahasch für die immer freundliche Unterstützung bei der Erstellung dieser Bachelor-Thesis, sowie der Hochschule Offenburg und der Fakultät Elektrotechnik und Informationstechnik.

Hiermit versichere ich eidesstattlich, dass die vorliegende Bachelor-Thesis von mir selbstständig und ohne unerlaubte fremde Hilfe angefertigt worden ist, insbesondere, dass ich alle Stellen, die wörtlich oder annähernd wörtlich oder dem Gedanken nach aus Veröffentlichungen, unveröffentlichten Unterlagen und Gesprächen entnommen worden sind, als solche an den entsprechenden Stellen innerhalb der Arbeit durch Zitate kenntlich gemacht habe, wobei in den Zitaten jeweils der Umfang der entnommenen Originalzitate kenntlich gemacht wurde. Ich bin mir bewusst, dass eine falsche Versicherung rechtliche Folgen haben wird.

(Datum, Unterschrift)

Inhaltsverzeichnis

1. EINLEITUNG 7

1.1 Motivation 7

1.2 Zielsetzung 8

1.3 Vorgehen 9

2. BUSINESS INTELLIGENCE UND DATA-WAREHOUSE 10

2.1 Motivation Business Intelligence 10

2.2 Definition Business Intelligence und Data Warehouse 10

2.3 Einsatzgebiete und Aufgaben 11

2.4 Referenzarchitektur Data Warehouse 12
2.4.1. Data-Warehouse-Manager 13
2.4.2. Datenquelle und Monitor 13
2.4.3. Quelldaten 13
2.4.4. Arbeitsbereich, Extraktion und Transformation 15
2.4.5. Data Warehouse 16
2.4.6. Analyse und Bericht 19

2.5 Umsetzung von Business Intelligence und Datawarehouse mit heutigen Datenbanksystemen 20
2.5.1. Multidimensionales Datenmodell 20
2.5.2. Technische Umsetzung des multidimensionalen Datenmodells 22
2.5.3. Probleme und Anforderungen 25

3. SPALTENORIENTIERTE UND IN-MEMORY DATENBANKSYSTEME 29

3.1 Spaltenorientierte Datenbanksysteme 29
3.1.1. Anordnung der Daten 29
3.1.2. Datenzugriff 30
3.1.3. Datenkompression 32

3.2 Spaltenorientierte In-Memory Datenbanksysteme 33

4. SAP HANA 35

4.1 Konzept und Architektur 36
4.1.1. Das Datenbanksystem: Spalten- und Zeilenorientiert? 36
4.1.2. Datenmodellierung mit SAP HANA Studio 37

4.2 Verschieden Versionen von SAP HANA 43
4.2.1. SAP HANA Standalone Version – SAP HANA 1.0 44
4.2.2. SAP HANA for SAP NetWeaver BW – SAP HANA 2.0 45
4.2.3. SAP HANA Softwareausführungen 45

4.3 Datensicherheit bei Stromausfall 46

4.4 SAP HANA für Business Intelligence 48
4.4.1. ETL Tools 48
4.4.2. Business Intelligence Tools 49

5. BEDEUTUNG VON SAP HANA FÜR BUSINESS INTELLIGENCE SYSTEME 51

5.1 Zusammenführung von OLTP und OLAP 51
5.1.1. In-Memory statt Festplatten 51
5.1.2. Zeilen- und Spaltenorientiertes Datenbanksysteme 52
5.1.3. Aufwendige Berechnungen auf Datenbankebene 52
5.1.4. Bedeutung für das Data Warehouse 53

5.2 Einsatzmöglichkeiten von SAP HANA außerhalb der „SAP Welt" 53
5.2.1. SAP HANA als Hauptpersistenz Schicht für Unternehmensanwendungen 54
5.2.2. SAP HANA als Data Warehouse 54
5.2.3. Eine Zusammenführung aus operativer Datenbank und Data Warehouse 55

5.3 Wie SAP HANA Business Intelligence verändern kann 55
5.3.1. SAP HANA als Data Warehouse im SAP Umfeld 55
5.3.2. SAP HANA als gemeinsame Persistenz für OLTP und OLAP 56

6. ANWENDUNGSSZENARIO 58

6.1 Beschreibung des gewählten Szenarios 58
6.1.1. Übersicht des Anwendungsszenarios 58
6.1.2. Architektur 60
6.1.3. Ziel der Anwendung 63

6.2 Bewertung 64
6.2.1. Kriterien 64
6.2.2. Schema 66

7. BUSINESS INTELLIGENCE MIT SAP HANA IN DER AMAZON CLOUD 67

7.1 Die Entwicklungsumgebung SAP HANA Studio 67
7.1.1. Einrichten des SAP HANA Studio 67
7.1.2. Datenbankentwicklung in der Cloud mit SAP HANA Studio 68
7.1.3. Probleme bei der Datenbankentwicklung mit SAP HANA Studio 70
7.1.4. Datenbankzugriff ohne SAP HANA Studio 72

7.2 ETL Prozesse 72
7.2.1. RSS Reader 73
7.2.2. RSS_StoreToHANA 74
7.2.3. Zug_Reader 75

7.3 Abbildung der Daten im SAP HANA Datenbanksystem 76
7.3.1. Procedures und SQLScript 77

7.4 Funktionsweise der Demoanwendung 79
7.4.1. Analytic- und Calculation Views in SAP HANA 80
7.4.2. Zugriff von Microsoft Excel 86

7.5 Bewertung der Anwendung in Bezug auf die Kriterien des Anwendungsszenarios 93
7.5.1. Cloudhosting 93

7.5.2. Vergleich der Entwicklungsumgebung mit klassischer Datenbankentwicklung 94
7.5.3. Dokumentation 95
7.5.4. Business Logik im Datenbanksystem 95
7.5.5. Realisierbarkeit der Demoanwendung 96

8. FAZIT 98

LITERATURVERZEICHNIS 100

ABBILDUNGSVERZEICHNIS 101

TABELLENVERZEICHNIS 102

LISTINGVERZEICHNIS 102

ABKÜRZUNGSVERZEICHNIS 102

1. Einleitung

Das SAP HANA Datenbanksystem im Kontext des Business Intelligence (BI) stellt das Kernthema dieser Thesis dar. Die nachfolgend erläuterten Konzepte und Systeme wurden daher immer auch unter dem Gesichtspunkt des Nutzens für das Themenfeld Business Intelligence erarbeitet.

1.1 Motivation

Ausgereifte BI Systeme liefern den Entscheidungsträgern in Unternehmen genau die Informationen, die sie in ihrer aktuellen Situation benötigen um daraus die richtigen Schlüsse zu ziehen. Darum ist Business Intelligence nach wie vor ein Top Thema in den Unternehmen.

Der kritische Erfolgsfaktor stellt dabei die Performance des Systems dar. Gerade bei großen Datenmengen kommen heutige Systeme schnell an Ihre Grenzen, was zur Folge hat, dass eine benötigte Information erst nach längerer Wartezeit verfügbar ist. In Zeiten von Cloud Diensten und mobilen Endgeräten, können diese Wartezeiten für Anwender inakzeptabel sein. Eine Ursache sind die verwendeten relationalen Datenbanksysteme. Datensätze werden dort Zeilenweise mit all deren Attributen aneinander gereiht im Speicher abgelegt, was den Lesevorgang über wenige Attribute aller (oder vieler) Datensätze langsam macht.

Spaltenorientierte Datenbanksysteme können aufgrund der internen Anordnung von Daten in Spalten die Performance von BI Anwendungen beschleunigen. Die Attribute sind im Speicher nicht mehr direkt durch Aneinanderreihung zu einem Datensatz zugeordnet. Vielmehr sind sie so angeordnet, dass gleichartige Attribute nacheinander im Speicher liegen. Somit kann die Lesegeschwindigkeit bei typischen BI Abfragen erheblich beschleunigt werden.

Eine weitere Performancesteigerung kann durch spaltenorientierte In-Memory Datenbanksysteme erreicht werden. Hierbei wird die Festplatte als Hauptspeichermedium aufgrund der langen Zugriffszeiten aus der Architektur eliminiert.

Alle Daten liegen zu jeder Zeit im Hauptspeicher des Systems, was aber erst durch immer mehr verfügbaren Hauptspeicher und Mehrkernprozessoren möglich wurde. Mehrkernprozessoren erlauben bei dieser Architektur auch das parallele verarbeiten von Daten.
Das in dieser Thesis behandelte System SAP HANA basiert auf einer solchen spaltenorientierten In-Memory Datenbank.

1.2 Zielsetzung

Im Rahmen dieser Thesis soll die Anwendbarkeit des SAP HANA Datenbanksystems in einer Cloud Umgebung auf BI Projekte analysiert werden. Die folgenden drei Themen werden hierbei besonders betrachtet.

Entwicklungsplattform in der Cloud

Die Entwicklung mit SAP HANA in der Cloud unterscheidet sich hauptsächlich darin, dass das Datenbanksystem vollständig ausgelagert ist. Hier wird eine Amazon Webservice Instanz verwendet. Eine lokale Entwicklung mit anschließendem Deployment ist daher ausgeschlossen. Ein Ziel dieser Arbeit ist die Beurteilung, welche Unterschiede, es im Vergleich zu klassischen Entwicklungsumgebungen gibt, bzw. wie Praktikabel diese Form der Entwicklung ist.

Entwicklung und Umsetzung einer Business Intelligence Anwendung

Durch die Entwicklung einer eigenen BI Anwendung basierend auf der SAP HANA Appliance soll die Entwicklungsumgebung einem Praxistest unterzogen werden. Die Anwendung verwendet die In-Memory Datenbank als Hauptdatenspeicher für Analysewerkzeuge.

Abschließende Bewertung zum Einsatz von SAP HANA

In den Unternehmen werden bereits BI Systeme eingesetzt. Für viele Unternehmen kann es jedoch aufgrund von Performanceproblemen interessant erscheinen eine In-Memory Datenbank wie SAP HANA in das bestehende System zu integrieren. Abschließend soll daher erörtert werden welchen nutzen das SAP HANA System als Komponente in Business Intelligence Systemen hat, wenn dabei auf weitere SAP Produkte verzichtet wird.

1.3 Vorgehen

Im folgenden Kapitel wird ein Überblick über das heutige Business Intelligence, insbesondere im Hinblick auf das Datenbanksystem, gegeben. Dies betrifft neben einem kurzen Blick auf den Nutzen und der Aufgaben im Unternehmensumfeld für BI-Systemen vor allem auch die technische Umsetzung mit derzeit verbreiteten Technologien und den daraus entstandenen Anforderungen und Problemen.
Resultierend aus den architektonischen Problemen aktueller Systeme, insbesondere im Bereich der Performance und der technologischen Weiterentwicklung der Hardwarekomponenten stellt die Ablöse der Festplatte als Speichermedium durch den Hauptspeicher eine erhebliche Optimierungsmöglichkeit dar. Kapitel 3 betrachtet das Prinzip von spaltenorientierten Datenbanksystemen in Verbindung mit der In-Memory-Technologie.
Mit den darauffolgenden Kapitel 4 und 5 wird die Umsetzung dieser Prinzipien anhand des SAP HANA Systems vorgestellt und auf die Anwendbarkeit und Bedeutung für das Business Intelligence analysiert.
In den letzten beiden Kapiteln ist die Planung, Durchführung und Bewertung einer Business Intelligence Demoanwendung mit SAP HANA dokumentiert. Es handelt sich dabei um eine Anwendung die verspätungsangaben der Deutschen Fernverkehrszüge sowie ausgewählte Wetter- und Nachrichtenfeeds in dem SAP HANA System verarbeitet. Die Anwendung soll vor allem zeigen, ob und wie SAP HANA als Plattform geeignet ist und bewerten wie sich die Entwicklung im Vergleich zu herkömmlichen Technologien dargestellt hat.

2. Business Intelligence und Data-Warehouse

Die Motivation für BI Projekte und die darauffolgenden Definitionen von Business Intelligence und dem Data-Warehouse (DWH) sollen einen Überblick über die Thematik verschaffen.

2.1 Motivation Business Intelligence

Durch den hohen Einsatz von IT-Systemen Unternehmen werden viele Daten generiert. Sie entstammen hauptsächlich aus den operativen Systemen. Hierbei handelt es sich beispielsweise um Geschäftsdaten wie Produktions-, Personal- und Absatzzahlen, technische Daten die von Sensoren gemeldet werden oder auch schwer maschinell lesbare Daten wie etwa Berichte von Außendienstmitarbeitern. Diese Daten werden auch als operative Daten bezeichnet. Hinzu kommen weitere externe Daten (z. B. von Online-Diensten).

Letztendlich sind diese angesammelten Daten jedoch wertlos, solange sie nicht zu verwertbaren Informationen aufbereitet werden können. Es ist die Aufgabe von Business Intelligence Anwendungen die nötige Datenaufbereitung zu realisieren, damit die Entscheidungsträger der Unternehmen aufgrund dieser Informationen die bestmöglichen Entscheidungen treffen können.

2.2 Definition Business Intelligence und Data Warehouse

Kemper definiert den Begriff Business Intelligence wie folgt:

„Unter Business Intelligence i. w. S. werden alle direkt und indirekt für die Entscheidungsunterstützung eingesetzten Anwendungen verstanden. Dieses beinhaltet neben der Auswertungs- und Präsentationsfunktionalität auch die Datenaufbereitung und -speicherung.“ [KBM2010, S. 4]

Das Data-Warehouse ist ein wichtiger Teilbereich des Business Intelligence, da es alle für das Business Intelligence benötigten Daten speichert. Erst die Erweiterung der *„Integrationsmöglichkeiten und die Hinzunahme des Wissensmanagements führt zum Business Intelligence.“* [BaGü2009, S. 14].

BaGü beschreibt das Data-Warehouse folglich als:

„Ein Data Warehouse ist eine physische Datenbank, die eine integrierte Sicht auf beliebige Daten zu Analysezwecken ermöglicht.“ [BaGü2009, S. 8]

2.3 Einsatzgebiete und Aufgaben

Die Implementierung von BI Projekten hat grundsätzlich immer dann eine Berechtigung, wenn aus einer großen Datenbasis Informationen gewonnen werden sollen, welche nicht aus dem täglichen Geschäftsbetrieb heraus ersichtlich sind. In BI Systemen werden Daten aus den transaktionsorientierten Datenbanksystemen (viele Schreib- und Lesevorgänge auf einzelne Datensätze) in analyseorientierte Datenbanksysteme (quasi ausschließlich lesender Zugriff auf eine Vielzahl von Datensätzen), innerhalb des Data-Warehouse, geladen. Durch diese Abgrenzung ist es den Anwendern von BI Systemen möglich, auf möglichst aktuellen Daten aufwendige Analyseanfragen auszuführen. Die entscheidenden Vorteile in dieser Abgrenzung liegen darin, dass zum einen die operativen Systeme nicht zusätzlich mit Anfragen belastet werden, und zum anderen, dass die Architektur des Data-Warehouse ausschließlich auf diese Art von Anfragen ausgerichtet ist und daher schneller Ergebnisse liefert.

Verwendet werden BI-Systeme in den Führungsebenen von Unternehmen, um mithilfe der aufbereiteten Informationen die operativen Ebenen zu steuern (vgl. Abb. 1).

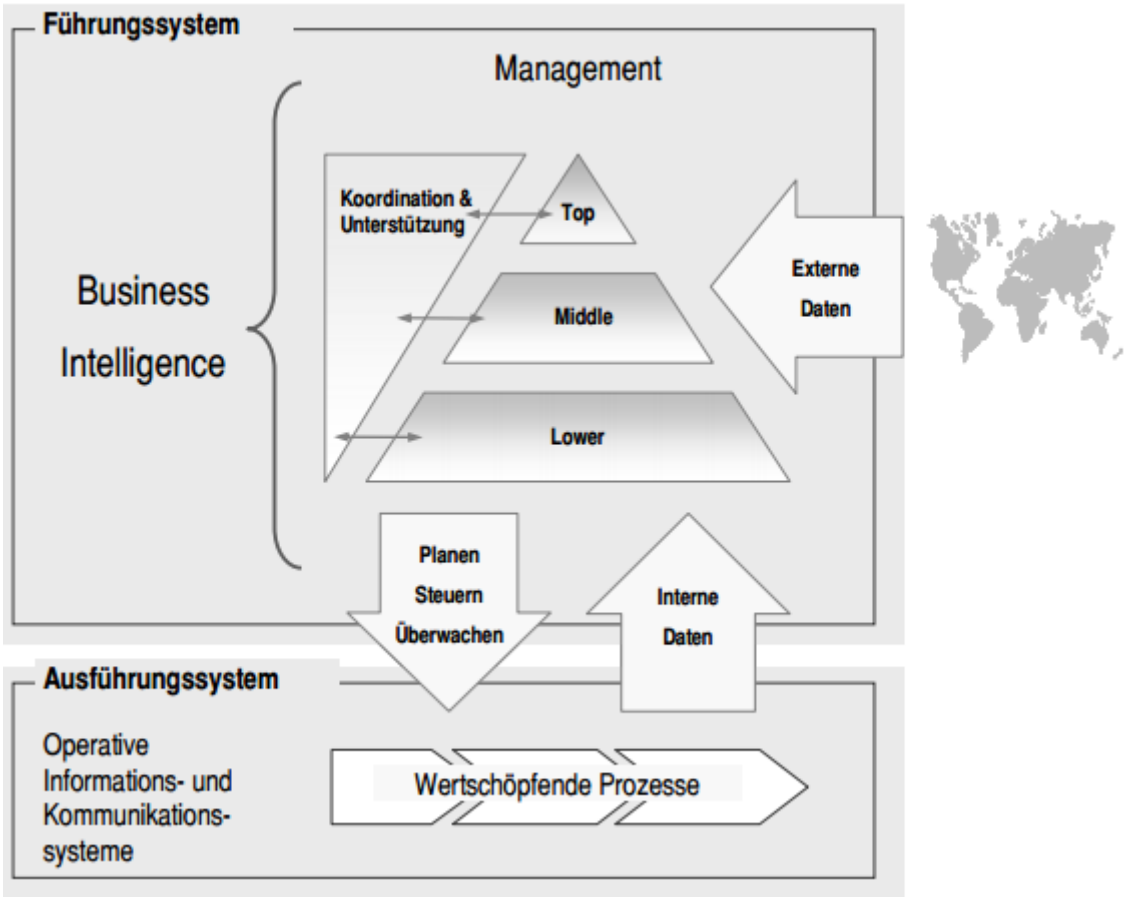


ABB. 1 BUSINESS INTELLIGENCE – ARCHITEKTUR [KBM2010]

Als Aufgaben, die mit Business Intelligence umgesetzt werden, gelten wie in Abb. 1 beschrieben neben der Überwachung und Steuerung durch die Analysewerkzeuge auch die strategische Planung. Da BI Systeme von den operativen Systemen abgekapselt sind, sind auch Planungsszenarien durch Manipulation der Daten im Data-Warehouse möglich.

Business Intelligence wird neben den betriebswirtschaftlichen Anwendungsgebieten auch in Wissenschaft (Klimabeobachtungen) und Technik (Messwertanalysen) eingesetzt.

2.4 Referenzarchitektur Data Warehouse

Zur allgemeingültigen Beschreibung eines DWH-System kann eine Referenzarchitektur verwendet werden. Die Referenzarchitektur stellt somit keine Norm oder Standard dar, sondern eine Grundlage an der sich DWH-Anwendungen orientieren.

Die im Folgenden vorgestellte Referenzarchitektur basiert auf [BaGü2009] und beschreibt das Data-Warehouse in einer technisch basierten Sicht.

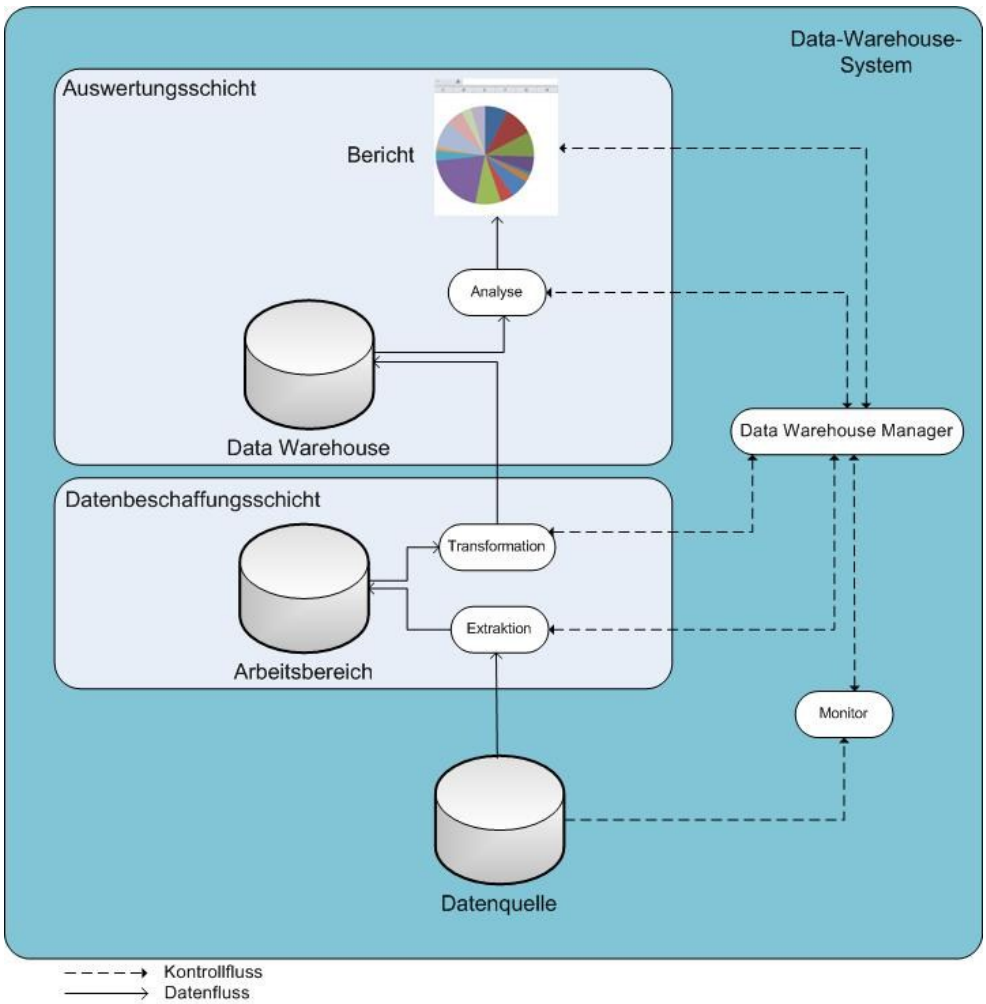


ABB. 2 REFERENZARCHITEKTUR NACH [BAGÜ2009]

2.4.1. Data-Warehouse-Manager

Der DWH-Manager stellt die zentrale Komponente des Data-Warehouse-System dar. Alle Prozesse, von der Extraktion der Daten aus den Datenquellen bis hin zur Analyse der Daten im Data-Warehouse werden mittels des DWH-Managers ausgelöst. Der Extraktionsprozesses durch den DWH-Manager kann dabei auf unterschiedliche Weise ausgelöst werden: In regelmäßigen Zeitabständen (z. B. täglich oder wöchentlich), abhängig von Datenänderungen welche durch die Monitoring Komponente übermittelt werden oder durch die explizite Extraktion durch einen Anwender.

2.4.2. Datenquelle und Monitor

Die Datenquelle, welche in der oben abgebildeten Referenzarchitektur als einzelne Datenbank abgebildet ist, kann in der Praxis häufig aus mehreren Quellen gebildet werden. Schon im internen Unternehmensumfeld sind verschiedenste Anwendungen im Einsatz, welche die operativen Daten unterschiedlich in ihren Datenbanksystemen speichern. Zudem können auch Externe Daten, etwa von Marktforschungsunternehmen oder Geschäftspartnern, mit zur Quelldatenbasis beitragen. Die Bewertung und Anforderung an Quelldaten wird daher im nachfolgenden Kapitel ausführlicher behandelt.

Wie schon im Abschnitt Data-Warehouse-Manager beschrieben wird der Monitor zum überwachen der Datenquellen genutzt, wobei jede Datenquelle typischerweise ihren eigenen Monitor hat. Die Hauptaufgabe der Monitore ist es Manipulationen in der Datenquelle zu identifizieren. Änderungen können beispielsweise durch Trigger oder Replikationsdienste von Datenbanksystemen erkannt werden. Stellt die Datenquelle selbst keine Mechanismen zur Propagierung von Manipulationen der Daten bereit so kann auch der aufwendige Vergleich von Snapshots zwischen Datenquelle und DWH Anwendung finden.

2.4.3. Quelldaten

Die internen und externen Daten bilden eine stark heterogene Quelldatenbasis, da zum einen die internen Daten von mehreren individuellen Anwendungen in eigenen Formatierungen gespeichert werden und zum anderen auf die Formatierung von externen Daten meist kein Einfluss genommen werden kann. So kann beispielsweise in

einem CRM System das Geschlecht der Kunden mit 0 (weiblich) und 1 (männlich) und im Personalverwaltungssystem mit w (weiblich) und m (männlich) codiert sein.
Diese Daten müssen daher in einem sogenannten Data-Warehouse integriert werden. In der Extraktionsphase des Data Warehousing wird zunächst festgelegt welche Datenquellen überhaupt in das Data-Warehouse integriert werden sollen, um anschließende diese Daten in den Arbeitsbereich des Data-Warehouse zu laden, wo sie schließlich in der Transformationsphase in ein einheitliches Schema überführt werden. [BaGü2009] .
Um in der Extraktionsphase, die richtigen Quelldaten auszuwählen, ist es somit unvermeidbar bei der Auswahl der Quelldaten ähnlich vorzugehen wie bei der Lieferantenauswahl. BaGü unterscheidet hierbei zwischen vier Kriterien:

Eignung zum Zweck

Die Daten sind nur geeignet, wenn sie auch zu dem Verwendungszweck des Data-Warehouse-Systems dienlich sind.

Die Qualität

Qualitätsmängel von Daten können z. B. sein:

- *„Inkorrekte Daten, verursacht durch Eingabe-, Mess- oder Verarbeitungsfehler*
- *Logisch widersprüchliche Daten*
- *Unvollständige, ungenaue bzw. zu grobe Daten*
- *Duplikate im Datenbestand*
- *Uneinheitlich repräsentierte Daten*
- *Veraltete Daten*
- *Für den Verwendungszweck irrelevante Daten*
- *Unverständliche Daten, bedingt durch qualitativ mangelhafte Metadaten“*

[BaGü2009, S. 42]

Die Verfügbarkeit

Selbst wenn bekannt ist, wo sich die benötigten Daten befinden, so ist vor allem bei externen Daten die Verfügbarkeit noch nicht gegeben. Die Verfügbarkeit von Quelldaten könnte durch organisatorische (z. B. Datenschutz) oder technische (z. B. Netzwerkzugriff) Voraussetzungen beschränkt sein.

Der Preis

Bei externen Daten kann zudem auch ein Preis erhoben werden, welcher gegen den Nutzen der Daten abgewogen wird. [BaGü2009]

2.4.4. Arbeitsbereich, Extraktion und Transformation

Der Arbeitsbereich ist die zentrale Komponente in der Datenbeschaffungsschicht. Diese Komponenten bilden zusammen mit der Ladekomponente den ETL-Prozess (ETL = Extract, Transform, Load).
Sämtliche aus den Datenquellen stammende Daten werden hier gespeichert. Die Extraktion wird durch den DWH-Manager gesteuert, wobei das Intervall der Extraktion abhängig von der gewählten Strategie ist.
Folgende Vorgehensweisen werden bei der Extraktion unterschieden (nach Kimball 2008 #6: 1–300})

Periodisch

Die Extraktionskomponente wird in festgelegten Zeitintervallen aktiviert. Hier werden Daten z. B. täglich, wöchentlich oder monatlich unabhängig davon ob diese sich auch geändert haben in das Data-Warehouse geladen werden.

Anfragegesteuert

Ein Anwender oder Administrator entscheidet wann Daten in das Data Warehouse übertragen werden sollen.

Ereignisgesteuert

Der Extraktionsprozess wird aufgrund von Ereignissen ausgelöst. Diese Variante ist jedoch nur anwendbar wenn die Datenquelle Ereignisse auslösen kann (wie z. B. ein Datenbanksystem).

Sofort

Ist eine hohe Aktualität gefordert (z. B. Börsenkurse), so kann auch jede einzelne Änderung in der Datenquelle in den Arbeitsbereich gelesen werden. Hierbei ist aber auf eventuelle Schwierigkeiten an der Performance zu achten.

Die Hauptaufgabe des Arbeitsbereiches ist die Integration der heterogenen Daten, welche durch die Transformationskomponente umgesetzt wird. Dies umfasst etwa die Anpassung oder Modifizierung von Datentypen, Kodierungen, Datumsangaben oder Maßeinheiten. Am Ende des ETL-Prozesses werden die Daten dann homogenisiert an das Data-Warehouse weitergegeben.

2.4.5. Data Warehouse

Das Data Warehouse, welches auf einem oder mehreren Datenbanksystemen aufbaut, dient als Basisdatenbank für die Analyse. Das Datenbankschema, sprich die Anordnung der Daten in logischen Tabellen und deren Referenzen, richtet sich ausschließlich an den Bedürfnissen der Analyse (vgl. 2.5.1 Multidimensionales Speichermodell).
Data-Marts bieten eine begrenzte Sicht auf das Data Warehouse. Sie enthalten eine Teilmenge der Daten, aus dem Data-Warehouse. Eine Abgrenzung kann zum Beispiel zwischen einzelnen Abteilungen erfolgen, sodass jede Abteilung ihre Analysen auf eigenen Data-Marts ausführt. Ein Data-Mart kann dabei physisch durch ein eigenes Datenbanksystem oder auch nur logisch von anderen Data-Marts abgegrenzt werden. Weiter wird zwischen abhängigen und unabhängigen Data Marts unterschieden. Im Falle der abhängigen Data Marts werden die Daten aus den ETL Prozessen in eine gemeinsame Basisdatenbank geladen.

Die Data Marts sind als Nabe-Speiche-Architektur (Abb. 2) um die Basisdatenbank des Data Warehause angeordnet. Die Data Marts erhalten aus der Basisdatenbank also nur Daten, welche für ihren speziellen Zweck zur Analyse aufbereitet werden sollen.

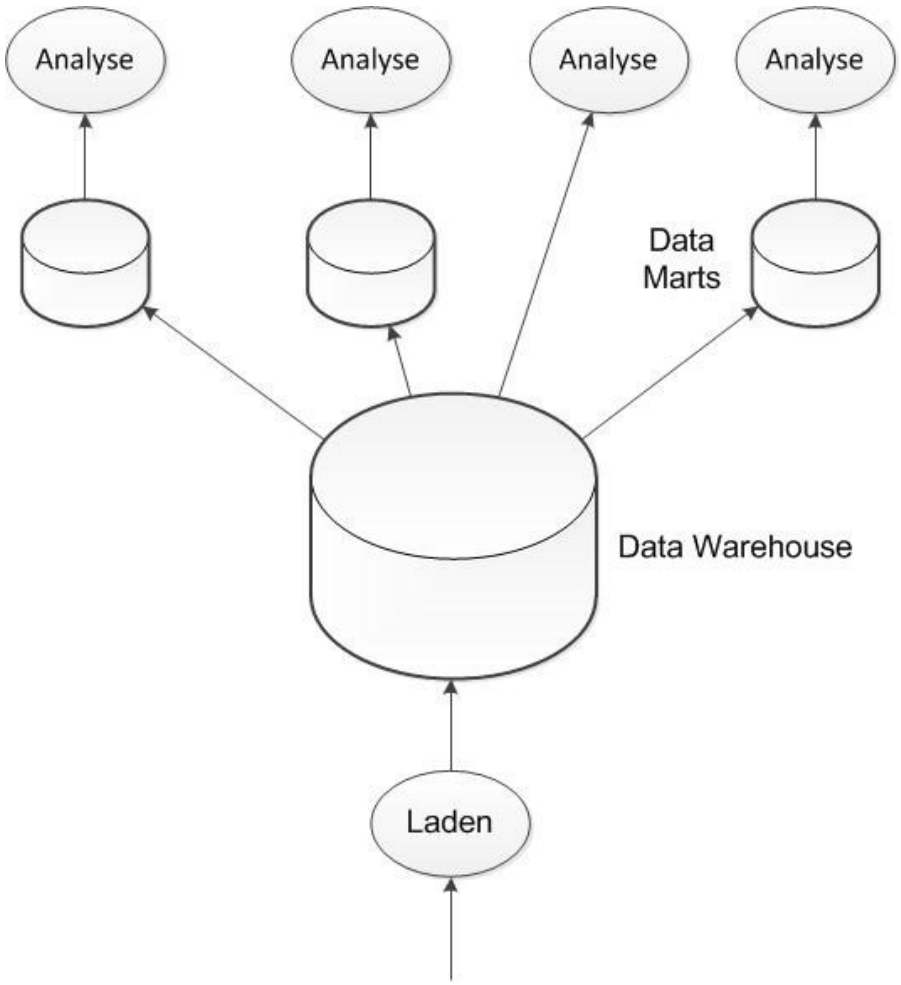


ABB. 2 ARCHITEKTUR BEI VERWENDUNG ABHÄNGER DATA MARTS. NACH [BAGÜ2009]

Bei unabhängigen Data Marts hingegen stellt jeder Data Mart sein eigenes DWH System dar. Diese Architektur entsteht vor allem dann, wenn verschiedene Unternehmensabteilungen eigene Data Warehouses einsetzen.

Der Vorteil dieser Architektur, in der die Basisdatenbank wie oben beschrieben entfällt, ist, dass auf diese Weise schon früh in DWH- Projekten brauchbare Resultate erzielt werden können. Dagegen sind aber beispielsweise abteilungsübergreifende Analysen schwerer umzusetzen. [BaGü2009]

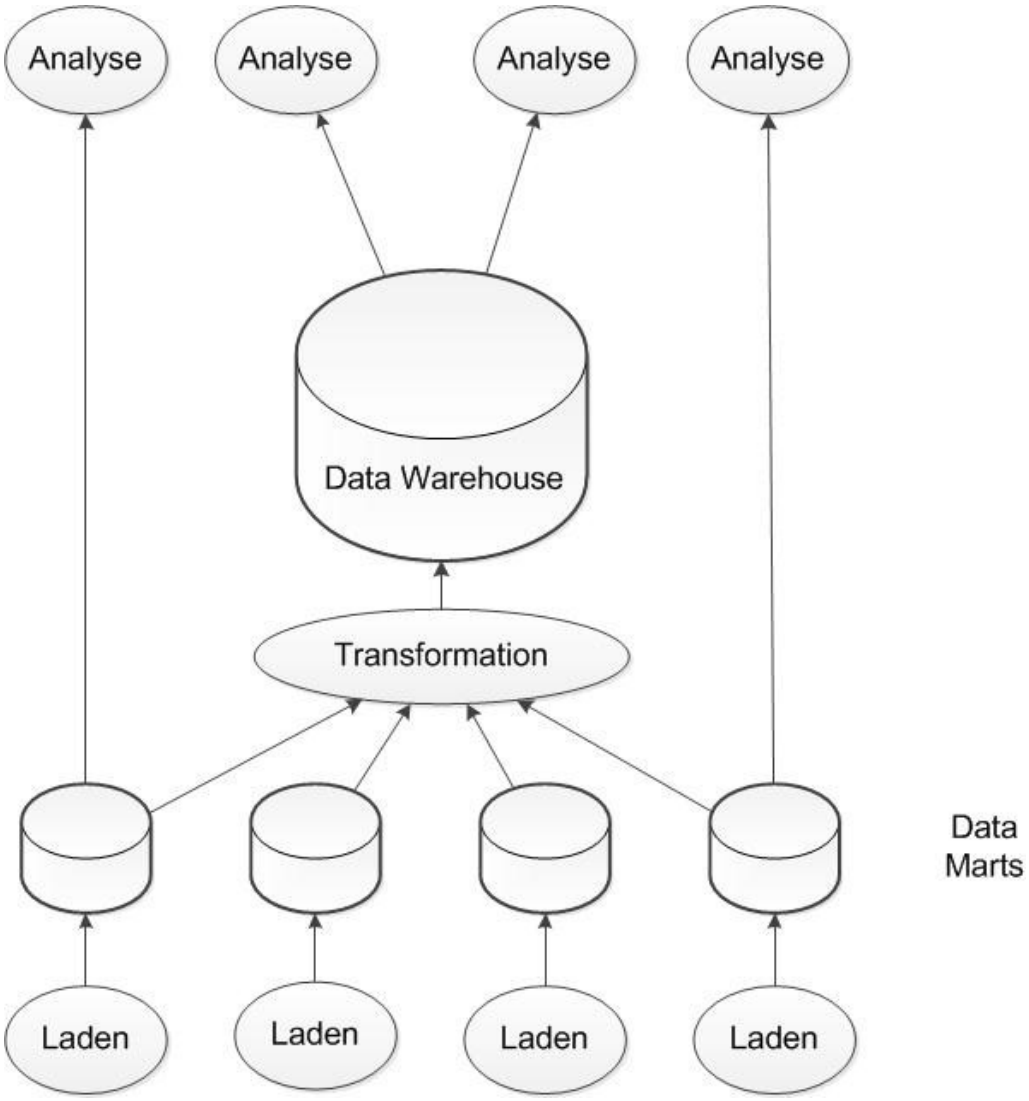


ABB. 3 ARCHITEKTUR BEI VERWENDUNG UNABHÄNGIGER DATA MARTS. NACH [BAGÜ2009]

2.4.6. Analyse und Bericht

„Die Anwenderaktzeptanz ist letztlich das entscheidende Kriterium für Erfolg oder Misserfolg jedes Data-Warehouse-Projektes. Alle technikzentrierten Fragen bezüglich Hardware, Datenbanksystem oder Modellierungsmethode sollten nicht um ihrer selbst willen, sondern mit den Erwartungen des Anwenders als ultimativen Anforderungsgeber behandelt werden." [BaGü2009, S. 66]

Der Auswertungsschicht mit den Komponenten für Analyse, Berichte kommt dabei eine entscheidende Rolle zu. Es sind die Schnittstellen zu den Anwendern des DWH-Systems. Vor allem das Vertrauen in die Ergebnisse der Analysen muss gegeben sein. Des Weiteren wird auch auf die Gestaltung der Berichte im Hinblick auf das schnelle erfassen der relevanten Informationen viel Wert gelegt.

Die Analyse realisiert die Generierung der vom Anwender gewünschten Informationen aus den Daten im Data Warehouse. *„Dies können einfach arithmetische Operationen (z. B. Aggregation) bis hin zu komplexen statistischen Untersuchungen (z. B. beim Data Mining) sein"* [BaGü2009, S. 66]. Die Auswertungen fließen außerdem zur schnellen Wiederverwendung oder für weitere Berechnungen in das Data Warehouse zurück.

Die Berichte, die aus den Analysen erstellt werden, können in verschiedenen Formen dargestellt werden. Die Wahl der Darstellungsform und des Mediums hängt dabei von den Zielanwendern ab.

Die Wahl der Zielplattform für Berichte wird dabei ein immer wichtigerer Faktor. Die Fat-Client Architektur bei der, der Computer des Anwenders selbst Berechnungen durchführt steht einer Thin-Client Architektur gegenüber. Der Client stellt nur noch die Darstellungsfunktion dar, weshalb aus technischer Sicht ein Webbrowser als Softwarekomponente am Client genügt. Alle Berechnungen werden im vorgelagerten DWH-System durchgeführt. Dies ermöglicht die breite Verteilung von Berichten an alle Managementebenen des Unternehmens. Durch aktuelle mobile Endgeräte sind Anwender auch fähig Ortsabhängig Berichte zu erhalten, was zu einem verstärkt flexiblen Business Intelligence führt. Um dies jedoch zu realisieren, muss eine performante Architektur zugrunde liegen, die Anfragen ohne große Verzögerungen bereitstellt. Das in dieser Arbeit untersuchte System SAP HANA soll nach Herstellerangaben diese Performanz bieten.

2.5 Umsetzung von Business Intelligence und Datawarehouse mit heutigen Datenbanksystemen

Dieser Abschnitt befasst sich mit der Frage der Datenhaltung im Data Warehouse, da hier im Gegensatz zu den operativen Systemen die Daten in multidimensionalen Modellen (vgl. Abschnitt 2.5.1) angeordnet sind, was jedoch nicht zwangsläufig die Verwendung eines alternativen Datenbanksystems bedeuten muss. Generell wird zwischen Online Analytical Processing (OLAP) und Online Transaction Processing (OLTP) unterschieden, wobei OLTP Datenbankabfragen und Schreibzugriffe in operativen Systemen bezeichnet und OLAP Datenbankabfragen über eine Vielzahl von Datensätze und Tabellen bezeichnet, was wiederum zur Verwendung des multidimensionalen Datenmodells führt.

2.5.1. Multidimensionales Datenmodell

Das multidimensionale Datenmodell wird nachstehend anhand eines Beispiels beschrieben. Eine typische Fragestellung, die für das Business Intelligence von Interesse sein könnte ist: „Wie hoch war der Umsatz einer Lebensmittelkette mit Bioerdbeeren im ersten Quartal 2012 für die Region Süddeutschland?“. Im Gegensatz dazu könnte eine OLTP Anfrage von einem Einkäufer einer Filiale wie folgt aussehen: „Wie viele Bioerdbeeren wurde im Februar 2012 verkauft?“. Der Hauptunterschied zwischen den beiden Anfragen stellen die im Multidimensionalen Datenmodell verwendeten Dimensionen dar. Die Anfrage des Einkäufers bezieht sich hier auf lediglich zwei Dimensionen (Produkt und Zeit). Die erste Fragestellung dagegen betrachtet drei Dimensionen (Produkt, Geographie und Zeit), wobei auch weitere Dimensionen, wie z. B. die Einschränkung auf einen bestimmten Erzeuger, hinzugefügt werden könnten. Weiterhin wird im Multidimensionalen Datenmodell von Hierarchien in den Dimensionen gesprochen. So stellt das Quartal in der Dimension Zeit eine Hierarchiestufe dar. Weitere Stufen könnten Tag, Monat und Jahr sein. Nachstehend wird das Konzept des OLAP Würfels, der diese Besonderheit grafisch darstellt, vorgestellt.

OLAP Würfel

Ein OLAP Würfel beschreibt eine Entität wie z. B. die Verkäufe. Die Dimensionen (hier: Produkt, Geographie und Zeit) stellen die Kanten des Würfels dar. Die Hierarchiestufen

können zudem an den Dimensionen angefügt werden. Abb. 4 zeigt einen OLAP Würfel der auch für obiges Beispiel gültig sein könnte.

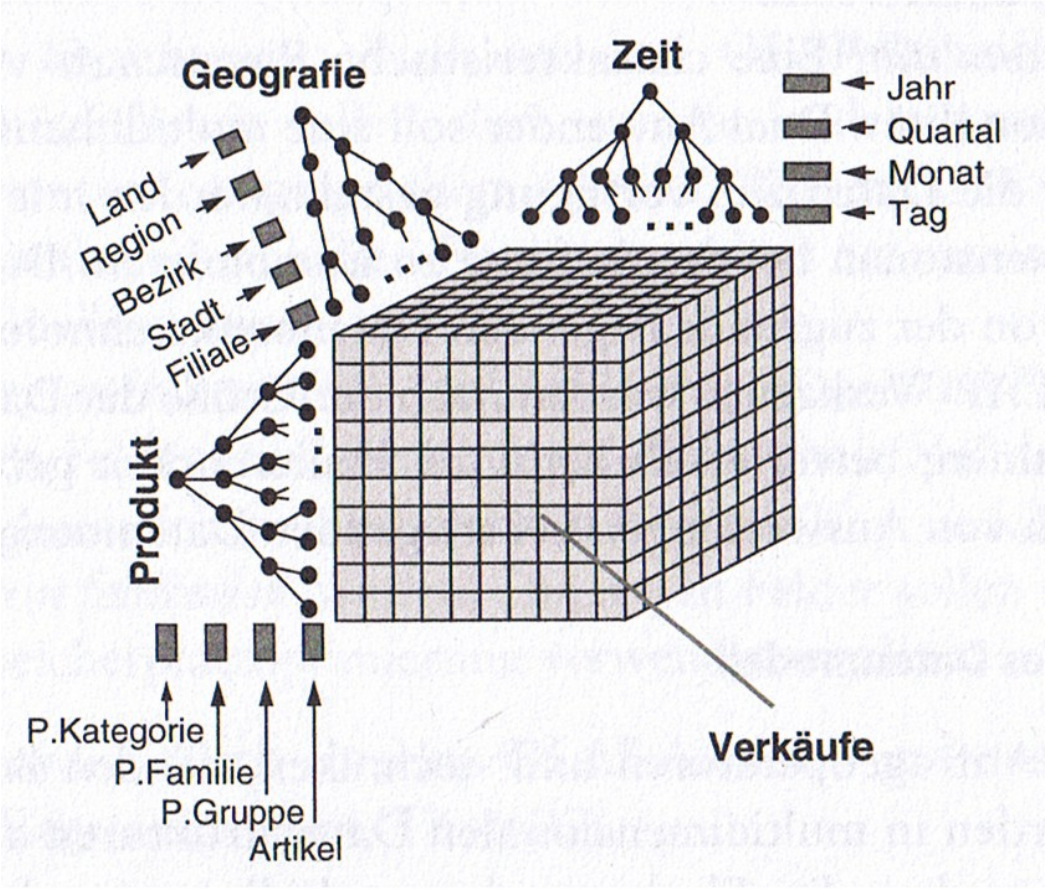


ABB. 4 OLAP WÜRFEL [BAGÜ2009]

Operationen für OLAP Würfel

Zur Verarbeitung von analytischen Anfragen im Data Warehouse, welche dem multidimensionalen Charakter von OLAP Würfeln entsprechen, wurden vor allem folgende OLAP spezifischen Operationen definiert:

- Pivotierung/Rotation
- Roll-up, Drill-down und Drill-across
- Slice und Dice

Diese müssen mittels einer Anfragesprache implementiert werden. In relationalen Datenbanksystemen ist dies typischerweise SQL (Structured Query Language). In Multidimensionalen Datenstrukturen finden häufig spezielle Sprachen wie MDX (Multidimensional Expressions) Anwendung.

Modellierung von multidimensionalen Datenstrukturen

Unabhängig von der physischen Implementierung auf Datenbanksystemen können multidimensionale Datenstrukturen grafisch dargestellt werden. Diese Darstellungsformen sind analog zu den für das relationale Datenmodell bekannten Modellen zu verstehen (etwas das Entity-Relationship-Model). Zur Modellierung von multidimensionalen Strukturen kann das multidimensionale Entity-Relationship-Mode (ME/R-Modell) verwendet werden [SBHD1998].

2.5.2. Technische Umsetzung des multidimensionalen Datenmodells

Das beschriebene Multidimensionale Datenmodell stellt lediglich die logische Anordnung der Daten dar. Zur technischen Umsetzung muss das Schema in einem Datenbanksystem umgesetzt werden. Die am stärksten verbreitete Architektur für Datenbanksysteme ist das Relationale Datenbanksystem, wobei Datensätze als Tabellen grafisch dargestellt werden. Die Umsetzung des multidimensionalen Modells auf dieses System wir im folgenden Abschnitt aufgezeigt. Die Verwendung von speziellen multidimensionalen Datenbanksystemen und hybride Formen werden anschließend erläutert.

Mit relationalen Datenbankensystemen (ROLAP)

Die Abbildung eines OLAP Würfels in einem relationalen Datenbanksystem funktioniert in einem ersten Schritt ohne größere Probleme. Eine Tabelle entspricht einer einzigen Entität und deren spalten entsprechen den Dimensionen des Würfels. Diese Tabelle wird Faktentabelle genannt. [BaGü2009]

Problematischer wird die Umsetzung schließlich durch die Miteinbeziehung von Hierarchien. Hierzu wurde das sogenannte Snowflake-Schema entwickelt. Jede Hierarchiestufe wie z. B. Tag, Monat und Jahr wird als eigenständige Relation modelliert. Diese Relationen enthalten neben ihren jeweiligen Primärschlüsseln insbesondere den Fremdschlüssel der darauffolgenden Relation welche die nächste Hierarchiestufe der Dimension darstellt. Der zusammengesetzte Primärschlüssel der Faktentabelle besteht aus den Fremdschlüsseln der verschiedenen Dimensionen. Die Faktentabelle enthält somit die Werte der Entität und die Ausprägung der Dimensionen wird durch den zusammengesetzten Primärschlüssel festgelegt.

Abbildung 5 zeigt das Snowflake-Schema für eine Entität Verkäufe mit den Dimensionen Produkt, Zeit und Geographie.

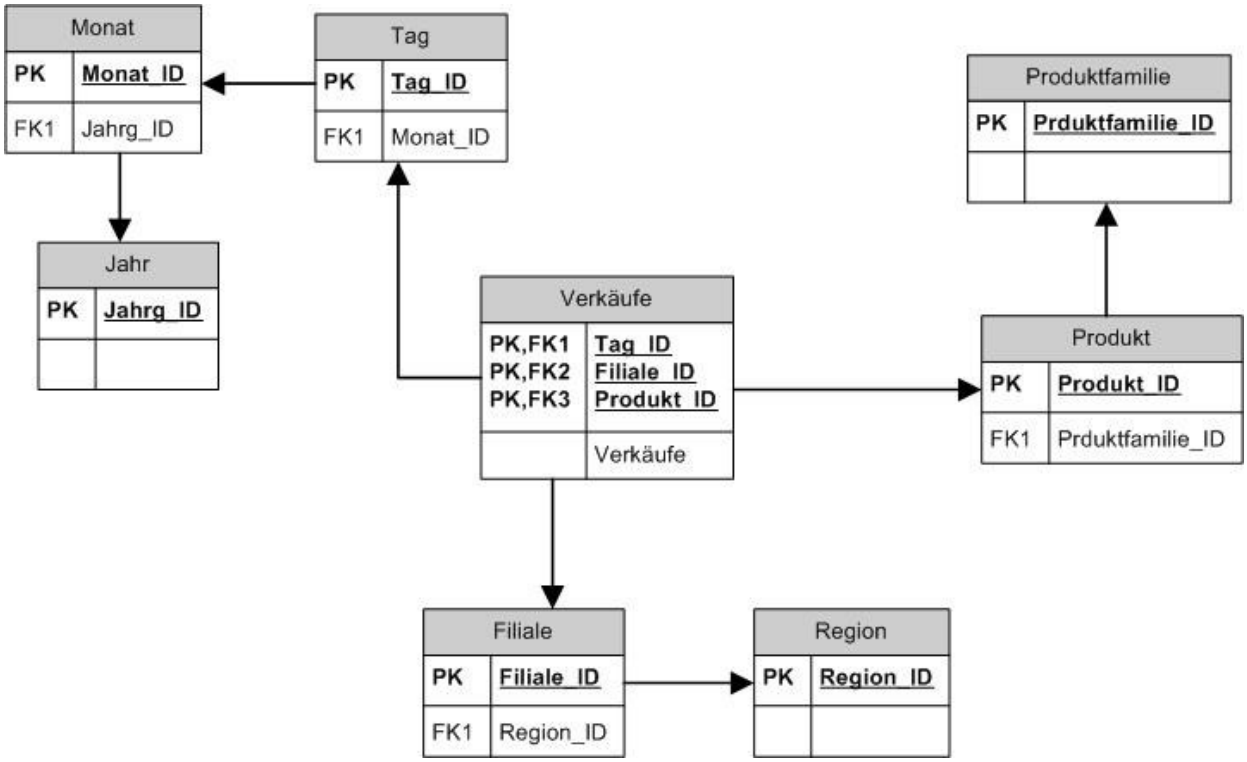


ABB. 5 SNOWFLAKE-SCHEMA

Der Vorteil dieser Anordnung ist, dass sie den Regeln der Normalisierung für relationale Datenbanksysteme entspricht und somit z. B. bei Änderungen an den Daten stabil ist. Ein großer Nachteil ist aber, dass bei einer multidimensionalen Anfrage eine Vielzahl von Tabellen in Form von Join-Operationen betroffen sind. Für die Datenbankmodellierung im Datawarehouse wird daher häufig auf die Vorteile der Normalisierung verzichtet.

Das de-normalisierte Schema bei dem die Hierarchiestufen lediglich als Attribute der Dimensionen bestehen wird dann als Star-Schema bezeichnet. Abb. 6 zeigt das aus Abb. 5 entwickelte Star-Schema.

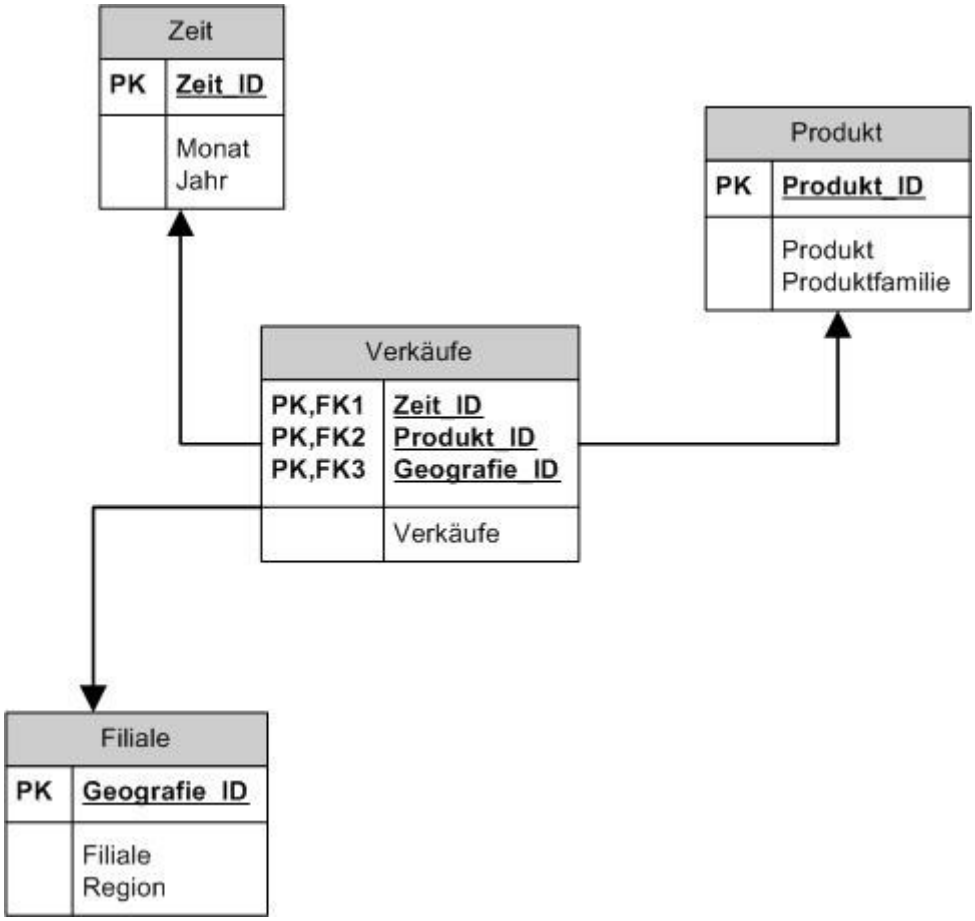


ABB. 6 STAR-SCHEMA

Auf multidimensionalen Datenbanksystemen (MOLAP)

Die Abbildung von multidimensionalen Datenstrukturen wie OLAP-Würfel in relationalen Datenbanksystemen ist aufgrund der starken Verbreitung von relationalen Datenbanksystemen ein funktionierender Ansatz, der aber aufgrund des Grundsätzlichen Unterschieds zum multidimensionalen Modell auch Nachteile mit sich bringt. So ist es beispielsweise nicht möglich multidimensionale Anfragen direkt auf das relationale DBMS anzuwenden, sondern muss mittels einer relationale Anfragesprache realisiert werden, was schnell zu sehr komplexen Ausdrücken führen kann.

Für multidimensionale Datenstrukturen wurden daher spezielle multidimensionale Datenbankmanagementsysteme (MDBMS) entwickelt. Sie ermöglichen es leichter

multidimensionale Aufgabenstellungen direkt mittels einer Abfragesprache zu formulieren (z. B. MDX).

OLAP-Würfel lassen sich direkter als bei Star- oder Snowflakeschema auf der Datenbank abbilden. Die physische Speicherung unterscheidet sich dabei grundsätzlich von relationalen DBMS. Die Dimensionen eines Würfels werden linearisiert und in einem Array gespeichert. Alle Dimensionen sind somit nacheinander angeordnet. Der logische Würfel ergibt sich schließlich durch Berechnung in Form von Offsets. Durch die Anzahl der Werte in einer Dimension ergibt sich die Länge einer Kante und somit der Offset für den Index an dem die nächste Dimension beginnt. Die Position einer Zelle des Würfels lässt sich somit berechnen und muss nicht, wie bei relationalen Datenbanksystemen, über mehrere Tabellen (und somit auch verschiedene Speicherbereiche) gebildet werden.

Hybride Formen

Um die Vorteile beider Datenbanksysteme (relational und multidimensional) nutzen zu können, gibt es auch Systeme die beides kombinieren. Das sogenannte Hybride OLAP (HOLAP) speichert Daten sowohl in relationalen als auch multidimensionalen Datenbanken, wobei in den relationalen Datenbanken eher Detaildaten und in den multidimensionalen eher aggregierte Daten zu finden sind. Der Benutzer greift in erster Linie auf das multidimensionale DBMS zu. Er kann daher intuitive multidimensionale Anfragen ausführen. Sind die angefragten Daten nicht im multidimensionalen DBMS verfügbar, werden die Daten aus dem relationalen DBMS generiert oder nachgeladen. Dies wird jedoch intern verarbeitet, sodass der Benutzer scheinbar auf einem MOLAP System arbeitet.

2.5.3. Probleme und Anforderungen

Die meisten aktuellen Business Intelligence Systeme basieren auf den zuvor beschriebenen Modellen. Die operativen Datenbanksysteme sind dabei stets relational und darauf ausgerichtet schnell auf einzelne Datensätze zuzugreifen, zu erstellen oder löschen. Business Intelligence Systeme können mit diesen operativen Daten nicht direkt arbeiten. Um diese Daten für Analysezwecke verfügbar zu machen wird das Data-Warehouse mit eigens entwickelten ETL-Prozessen befüllt. Die Daten als gesamtes werden hierbei von ihrer relationalen Anordnung in einen multidimensionalen Raum

überführt. Im Folgenden werden einige Probleme und Anforderungen beschrieben mit denen Business Intelligence heute konfrontiert ist.

Trennung von OLTP und OLAP

Die Trennung von OLTP und OLAP Systemen liegt vor allem an deren unterschiedlicher Charakteristik wie diese mit Daten umgehen. Folgende Tabelle aus [PlZe2011] macht dies deutlich.

Operational Processing	**Analytical Processing**
Pre-determined queries	Ad-hoc queries
Simple queries	Complex queries
Highly selective query terms, small found sets	Low selectivity, large found sets
Real-time updates	Batch inserts and updates
Inserts, updates and selects	Mainly selects
Short transaction runtimes	Long transaction runtimes
Large number of concurrent users (1000+), large number of transactions	Small number of concurrent users (100+), few transactions

TABELLE 1 VERGLEICH OLTP UND OLAP NACH [PLZE2011]

Die Tabelle suggeriert, dass die Verarbeitung von Daten in OLAP und OLTP Systemen so wenig gemeinsam haben, sodass es schon deshalb eigene Datenbanksysteme für den jeweiligen Anwendungsfall geben muss. Leseorientierte Datenbanksysteme für OLAP Anwendungen und schreiborientierte Datenbanksystem für OLTP Anwendungen.

Eine Studie des Hasso Plattner Instituts über 65 Installationen eines ERP Systems (Abb. 7) zeigt jedoch auf, dass der unterschied längst nicht so groß wie der TPC-C Benchmark vermuten lässt.

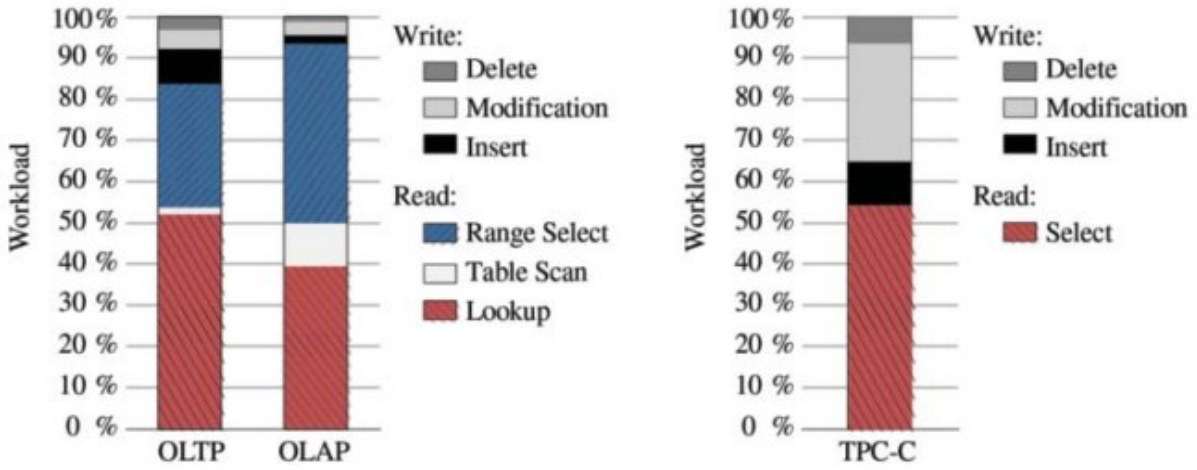


ABB. 7 WORKLOADS NACH ANWENDUNGSFALL IM VERGLEICH ZUM TPC-C BENCHMARK [KRJO2012]

Vor allem der Anteil an Schreibzugriffen spielt in ERP Systemen nur eine untergeordnete Rolle.

Die Eigenschaft des Data-Warehouses als integrierter Speicher für alle internen und externen Daten, sowie deren Historisierung ist ein weiterer Grund für die Trennung der Systeme. Durch die Abkapselung der Daten führen die immensen Datenvolumen eines Data-Warehouse nicht zu Performance Engpässen in den operativen Systemen. Hinzu kommt, dass eine direkte Abfrage eines Analysetools über mehrere Datenquellen nur schwer umsetzbar wäre, da innerhalb der Abfrage eine aufwendige Vereinheitlichung der Daten durchgeführt werden müsste.

Andererseits wir durch die Trennung die gemeinsame Nutzung der Daten erschwert. So ist nicht auszuschließen, dass nicht auch Rohdaten eines operativen Systems im Business Intelligence Kontext benötigt werden oder operative Anwendungen mit Informationen aus dem Data-Warehouse versorgt werden sollen. [PlZe2011]

Mangelnde Performanz

Durch die zuvor beschriebene Trennung von operativen und analytischen Daten ergeben sich im Zusammenhang mit zunehmenden Datenströmen weitere Engpässe.

In vielen Unternehmen werden mittlerweile nicht mehr nur interne, betriebswirtschaftliche Daten in die Business Intelligence Systeme geladen. Banken

und andere Unternehmen aus dem Bereich Finanzen greifen auf externe Daten wie Aktienkurse zu. In Industriebetrieben werden beispielsweise Daten aus Produktionsabteilungen miteinbezogen: Produzierte Mengen, Werte von Maschinensensoren, Ausfallzeiten und Fehlerberichte. Im Zuge der Vernetzung von Kunden und Lieferanten zur Optimierung von Supply Chains werden auch solche Fremddaten wie Lagerbestände und Auftragsdaten ins eigene System integriert.

Will ein Unternehmen alle seine gesammelten Daten nun auch noch qualitativ verarbeiten um auch einen wirtschaftlichen Nutzen zu generieren, sind aufwendige Berechnungen auf großen Datenmengen nötig. Anfragen auf diese Daten können dann mit heutigen Architekturen unter Umständen auch mehrere Minuten bis Stunden berechnen.

Ein gängiger Ansatz zur Beschleunigung dieser Vorgänge ist die Bildung von Aggregaten. Viele gleichartige detaillierte Daten werden konsolidiert und als einzelne Kennzahl im Data-Warehouse gespeichert. Dies könnte zum Beispiel die Darstellung aller getätigten Umsätze eines Tages in einer Zahl sein. Dabei gehen jedoch auch Detailinformationen verloren. Die Frage nach dem höchsten getätigten Einzelumsatz eines Tages gehen in vorangegangenem Beispiel verloren oder müsste als weitere aggregierte Kennzahl gespeichert werden.

Neue Technologien als Lösung?

Laut PlZe ist die Trennung von analytischen und operativen Systemen aus technischer Sicht der Vergangenheit begründet, wobei die Integration von Daten im Data-Warehouse eine Ausnahme darstellt. Aufgrund der technologischen Entwicklungen der letzten Jahre sei es nun wieder Möglich Business Intelligence auf operativen Systemen anzuwenden. (*“As explained previously, the principal reasons to separate analytical from operational systems were technical. An exception was the need to consolidate complex, heterogeneous system landscapes. As a result of the technological developments in recent years, many technical problems have been solved. We propose that BI using operational data could be once again performed on the operational system.“* [PlZe2011, S. 183]).

Spaltenorientierte Datenbanksysteme im Hauptspeicher in Verbindung mit der parallelen Nutzung von mehreren CPU Kernen sind diese angesprochenen neuen technologischen Möglichkeiten, die im folgenden Kapitel beschrieben werden und in der SAP HANA Appliance (Kapitel 4) Anwendung finden.

3. Spaltenorientierte und In-Memory Datenbanksysteme

Spaltenorientierte Datenbanksysteme stellen schon alleine durch eine andere Anordnung der Daten im Gegensatz zu relationalen Datenbanksystemen einige Vorteile im Bereich Business Intelligence dar, welche später in diesem Kapitel beschrieben werden. Zuvor wird im nächsten Abschnitt 3.1 das Prinzip von Spaltenorientierten Datenbanksystemen erläutert.

Im Zusammenhang mit der In-Memory Technologie (Abschnitt 3.2), welche davon ausgeht dass alle Daten jederzeit im Hauptspeicher für eine Verarbeitung verfügbar sind, stellt dies für das Business Intelligence eine erhebliche Performancesteigerung dar.

3.1 Spaltenorientierte Datenbanksysteme

In Spaltenorientierten Datenbanksysteme werden Daten, wie die Bezeichnung vermuten lässt, in Spalten angeordnet. Das wichtigste Merkmal dieser Architektur ist daher, dass die nicht Daten eines einzelnen Datensatzes (Tupel) aneinander gereiht auf dem Datenträger abgelegt werden, sondern es werden die einzelnen Spalten, also die Attribute, aller Tupel aneinander gereiht abgelegt. Folglich benötig daher in einer spaltenorientierten Datenbank jedes Attribut auch einen zusätzlichen Index um einem Datensatz zugeordnet zu werden, was jedoch vernachlässigt werden kann, da dies innerhalb des Datenbanksystems gelöst wird und nur geringe Auswirkungen auf die Performance hat.. Die entscheidenden Merkmale des Systems, die es von relationalen Systemen unterscheiden, befinden sich innerhalb des Datenbanksystems, sodass auch spaltenorientierte Datenbanksysteme mit gängigen Abfragesprachen wie SQL und MDX ansprechbar sind. Die Wichtigsten Merkmale wie die Anordnung der Daten, den internen Datenzugriff, Datenkompression und Parallelisierung sind im Folgenden beschrieben.

3.1.1. Anordnung der Daten

Abb. 8 zeigt die Anordnung von Daten im Speicher in einer Spaltenorientierten Datenbank im Vergleich zur relationalen Anordnung. Zur Veranschaulichung wurde eine Tabelle *ADRESSE* gewählt mit den Attributen *ADRESSEID*, *STRAßE*, *HAUSNUMMER*, *POSTLEITZAHL* und *LAND*.

Auf Basis dieser Tabelle wird dargestellt wie vier Datensätze im Speicher einer relationalen und spaltenorientierten Datenbank angeordnet werden.

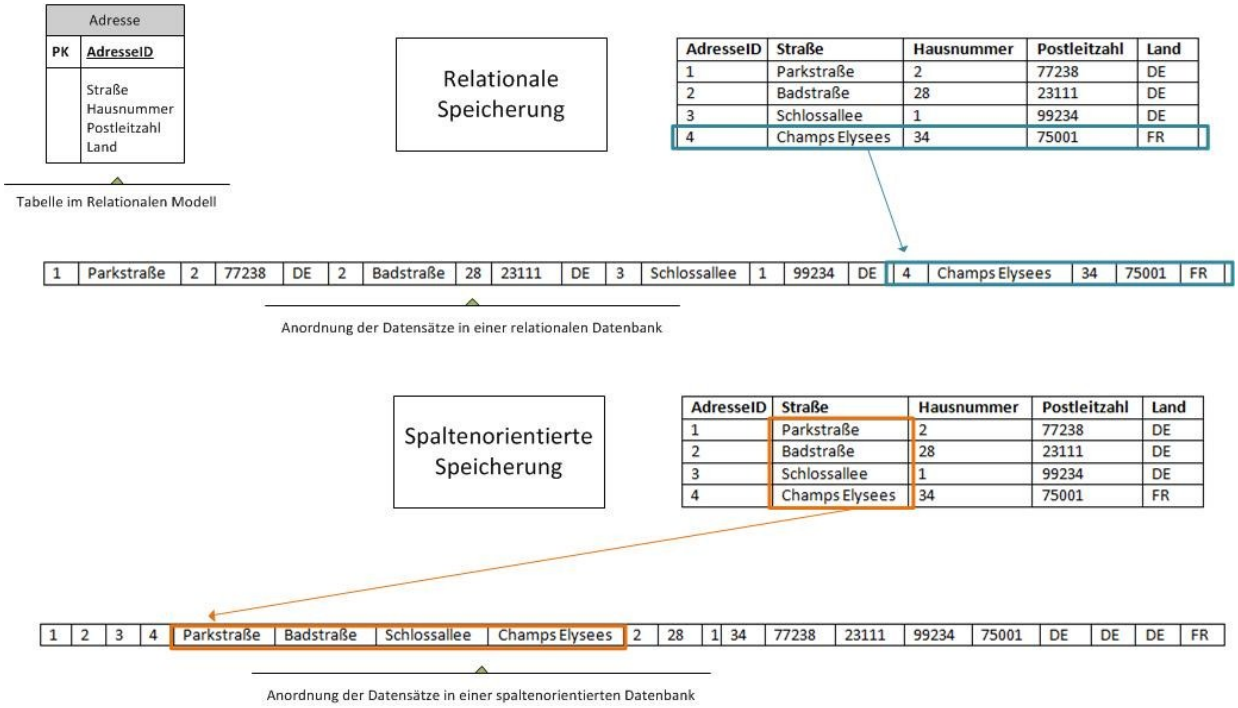


ABB. 8 ANORDNUNG VON DATEN IM SPALTENORIENTIERTEN DATENBANKSYSTEMEN

In der obigen Abbildung ist zu sehen, dass die Daten im Speicher linear angeordnet sind. Bei der relationalen Speicherung wird somit ein Tupel am Stück im Speicher abgelegt. Ein Datensatz folgt somit auf den nächsten. In der spaltenorientierten Speicherung werden dagegen alle Werte eines Attributes aller Datensätze gespeichert. Daraus ergeben sich entscheidende Vorteile beim Datenzugriff und der Datenkompression.

3.1.2. Datenzugriff

Der Zugriff auf Daten in spaltenorientierten Datenbanksystemen ist für den Anwendungsentwickler kaum verschieden zu einem relationalen Datenbanksystem. Er kann auch hier die Abfragesprache SQL nutzen. Aufgrund der unterschiedlichen Anordnung der Daten ergeben sich aber Datenbankintern auch Unterschiede beim Datenzugriff. Darauf wird mit folgenden Beispielen eingegangen.

Tabelle 2 zeigt Beispieldatensätze wie sie in einer Datenbank ausgehend von dem Eingangsbeispiel dieses Kapitels gespeichert sein könnten. In Abb. 9 ist die zugehörige Anordnung der Daten in einem spaltenorientierten Datenbanksystem zu sehen.

AdresseID	Straße	Hausnummer	Postleitzahl	Land
1	Parkstraße	2	77238	DE
2	Badstraße	28	23111	DE
3	Schlossallee	1	99234	DE
4	Champs Elysees	34	75001	FR

TABELLE 2 DATENSÄTZE DER BEISPIELTABELLE

1	2	3	4	Parkstraße	Badstraße	Schlossallee	Champs Elysees	2	28	1	34	77238	23111	99234	75001	DE	DE	DE	FR

ABB. 9 ANORDNUNG DER DATEN IN EINEM SPALTENORIENTIERTEN DBMS

Zugriff auf einzelnen Datensatz

Soll ein vollständiger Datensatz gelesen werden, etwa zum Anzeigen der Adressdaten eines Kunden, so muss beim Zugriff auf die Daten gesprungen werden, da diese nicht, wie bei der relationalen Speicherung, in Reihe im Speicher liegen. Diese Art von Anfragen benötigen somit in spaltenorientierten DBMS länger als in relationalen DBMS, weshalb für operative Systeme mit OLTP Ansatz das relationale Modell als geeigneter scheint (vgl. Abschnitt 2.5.3). Ebenso ist auch beim Einfügen von neuen Datensätzen das relationale Datenbanksystem geeigneter.

Zugriff auf einzelnes Attribut mehrerer Datensätze

Erst wenn man das Datenbanksystem in einem analytischen Kontext wie dem Business Intelligence betrachtet, werden die Vorteile der spaltenorientierten Anordnung deutlich. Soll für das Eingangsbeispiel etwa ermittelst werden wie viele Adressen sich in Deutschland befinden, so muss nur das Attribut Land gelesen werden, welche am Stück im Speicher liegen. Viel bedeutender ist jedoch die Tatsache, dass hier nur eine von fünf Spalten gelesen werden muss. Die gleiche Anfrage bei relationaler Speicherung hätte das vollständige lesen aller Datensätze benötigt. [PlZe2011]

Abb. 10 vergleicht nochmals schematisch den Zugriff auf Daten in relationalen- und spaltenorientierten Datenbanksystemen.

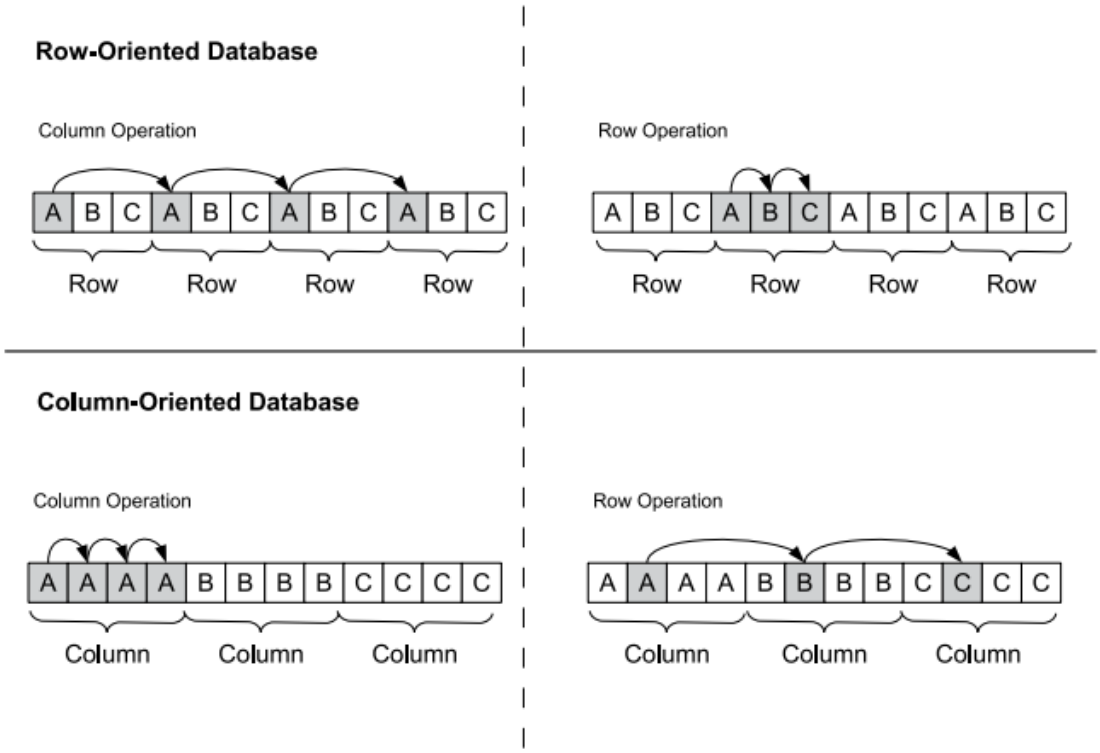


ABB. 10 VERGLEICH DES ZUGRIFFS BEI RELATIONALEN- UND SPALTENORIENTIERTEN DBMS[PLZE2011]

3.1.3. Datenkompression

In einer spaltenorientierten Datenbank, sind Kompressionsalgorithmen besonders effektiv, was vor allem daran liegt, dass die Daten, wie in Abb. 10, nach Attributen aufgereiht im Speicher liegen. Es können also ganze Spalten komprimiert werden anstelle der logischen Datensätze, welche aus verschiedenen Datentypen zusammengesetzt sind. Zwei besonders für Spaltenorientierte Datenbanken geeignete Verfahren werden hier beispielhaft dargestellt.

Dictionary Encoding

Beim Dictionary Encoding werden die verschiedenen Ausprägungen eines Attributes in einem Verzeichnis gespeichert und die Ausprägungen selbst durch einen Index ersetzt. Auf diese Weise wird die eigentliche Ausprägung, welche zum Beispiel ein Wort mit zehn Zeichen sein könnte, auf eine einzige Zahl reduziert. [PlZe2011]

Common Value Suppression

Wenn in einer Spalte häufig dieselbe Merkmalsausprägung vorkommt, wie beispielsweise ein „Null-Wert“, können diese gewöhnlichen Werte (Engl.: Common Value) besonders komprimiert werden. Bei häufiger Aneinanderreihung derselben Merkmalsausprägung werden mit Prefix Coding beim ersten Eintrag die folgenden lediglich durch die Anzahl der Wiederholungen ersetzt. Das Sparse Coding dagegen ist effizient, wenn eine häufig auftretende Merkmalsausprägung über die gesamte Spalte verteilt ist. Dazu wird jede Merkmalsausprägung der Spalte mit einem Extra Bit versehen, welches angibt ob es sich bei der Merkmalsausprägung um die häufig vorkommende (wie etwa dem „Null-Wert“) oder eine Verschiedene handelt. Die Merkmalsausprägung muss in einer Abfrage also nur gelesen werden, wenn es sich um einen Ausnahmewert handelt. [PlZe2011].

3.2 Spaltenorientierte In-Memory Datenbanksysteme

Spaltenorientierte In-Memory Datenbanksysteme mach sich die Vorteile der spaltenorientierten Datenbanksysteme zu nutzen und optimieren diese um das Konzept der In-Memory Technologie. Mit der In-Memory Technologie ist die vollständige Abbildung aller Daten im Hauptspeicher eines Systems gemeint. Festplatten dienen lediglich der Sicherung und Archivierung der Daten.

Parallelisierung

Die technische Entwicklung im Bereich der Prozessoren hat sich seit dem Jahr 2001 weg von der stetigen Steigerung der Taktfrequenz hin zu der Erweiterung der Prozessoren um mehrere Kerne geändert [PlZe2011]. Diese Entwicklung hat zu Folge, dass Anwendungen nur einen Nutzen daraus erzielen können, wenn sie auch auf mehreren Rechenkernen gleichzeitig verarbeitet werden können.

Ein Spaltenorientiertes In-Memory Datenbanksystem, welches auf Servern mit einer Vielzahl von Serverblades aufsetzt kann mit der Ausführung von Prozessen auf jeweils ihren eigenen Rechenkernen deutlich höhere Performanceraten erreichen. Gerade auch weil keine Daten von Festplatten gelesen werden müssen.

SanssouciDB ist ein theoretischer Entwurf eines solchen spaltenorientierten Datenbanksystems welches am Hasso Plattner Institut in Potsdam entwickelt wurde. Es diente als Vorlage des in Kapitel 4 vorgestellten SAP HANA Datenbanksystems und wendet in folgende Fällen spezielle Parallelisierungsalgorithmen an.

Lesevorgänge

Bei Lesevorgängen über mehrere Spalten wird in jede Spalte von einem eigenen Thread gelesen. Die Ergebnisse werden dann am Ende wieder zusammengeführt.

Aggregationen

Die Aggregation von Daten wird in der Regel durch Hash-Tabellen implementiert. Spezielle Hash-Tabellen Algorithmen erlauben das parallele einfügen von Daten ohne die komplette Tabelle sperren zu müssen. In SanssouciDB können mehrere Threads in diese Hash-Tabelle parallel schreiben.

Joins

Mithilfe von Joins werden Daten aus zwei oder mehreren Tabellen einer Datenbank zusammenfasst. Auch hier werden Hash-Tabellen verwendet die von SanssouciDB parallel verarbeitet werden können.
[PlZe2011]

4. SAP HANA

SAP HANA ist eine Appliance, also die Verbindung von Hard- und Softwarekomponenten in einem Produkt, entwickelt von SAP. Die SAP präsentiert SAP HANA in [Word2012] als eine „für die Enterprise IT Industrie massiv eingreifende Innovation (*„SAP HANA is one of those massively disruptive innovations for the enterprise IT industry.“* [Word2012, S. 15]) und vergleicht das Produkt mit der Entwicklung des E-Books 500 Jahre nach der Erfindung des Buchdruckes. Verglichen wird dabei die Loslösung vom Papier als Medium für Bücher mit der Festplatte als Medium für Datenbanksysteme. Der simple Grund die Festplatte zu ersetzten ist dabei, dass die Zugriffszeiten auf Daten die auf einer Festplatte gespeichert sind signifikant höher sind als bei Daten die sich im Hauptspeicher befinden. Das Konzept von SAP HANA ist somit die Eliminierung der Festplatte aus der Kernarchitektur, was heutzutage aufgrund der geringen Kosten für Hauptspeicher möglich ist. Die Festplatte soll lediglich als Archivmedium dienen. Dieser Ansatz bestätigt somit eine Vorhersage bezüglich Speichermedien von Jim Gray aus dem Jahr 2006:

- *„Tape is Dead*
- *Disk is Tape*
- *Flash is Disk*
- *RAM Locality is King“*

[SIGG2012 S. 17].

Folglich ist das Datenbanksystem der SAP HANA Appliance als echte In-Memory Datenbank konzipiert. Alle Daten in der Datenbank sind zu jeder Zeit im Hauptspeicher verfügbar. Die enge Verzahnung mit der Hardware als Appliance stellt dabei die notwendige Kapazität sicher. Konkret bedeutet dies Hauptspeichergrößen bis in den Terrabyte Bereich und annähernd 100 Rechenkerne [Word2012]. SAP HANA kann diese Leistung vor allem auch dazu nutzen analytische Berechnung direkt auf den Daten auszuführen. Gegenüber der Ausführung solcher aufwendigen Berechnung in einer aufgesetzten Anwendungsschicht, entlastet die datenbankseitige Ausführung die darauf zugreifenden Anwendungssysteme, welche somit eventuell selbst mit weniger Leistung auskommen.

Ein Merkmal, welches für SAP wohl schon aufgrund von Vertriebsgründen forciert wurde, ist eine möglichst einfache Integration von SAP HANA in bestehende SAP Anwendungssysteme [Word2012], was in dieser Arbeit jedoch nicht näher betrachtet wird. Die SAP HANA Datenbank ist auch eine voll funktionsfähige SQL – basierte Datenbank.

4.1 Konzept und Architektur

SAP HANA ist im Kern ein Datenbanksystem, was gerade auch in [Berg2012] ausdrücklich betont wird. Dennoch gehen die Einsatzmöglichkeiten von SAP HANA über die Fähigkeiten von klassischen relationalen- oder spaltenorientierten Datenbanksystemen hinaus. Die Grundlage dafür bietet die In-Memory Technologie und die Tatsache, dass SAP HANA eng mit der Hardware als Appliance vertrieben wird. Von dieser Vorgehensweise wird jedoch mit der Cloudlösung SAP HANA ONE teilweise abgewichen, da diese als Instanz bei Amazon Webservices gehostet ist. So ist sichergestellt, dass jedes SAP HANA System die Performance Anforderungen erfüllt. Die wesentlichen Merkmale des Systems werden in den folgenden Abschnitten erläutert.

4.1.1. Das Datenbanksystem: Spalten- und Zeilenorientiert?

Die physische Speicherung von Daten erfolgt in SAP HANA mit spaltenorientierter Logik. Die Daten können somit intern stark komprimiert werden. Im Speziellen werden unter anderem folgende Komprimierungsalgorithmen verwendet: *„prefix coding, run length coding, cluster coding, sparse coding and indirect coding“* [Berg2012, S. 136]. Des Weiteren ist die Spaltenorientierung für analytische Anfragen wesentlich besser geeignet (Abschnitt 3.1) als die zeilenorientierte Speicherung der Daten.

Um jedoch sowohl transaktionalen als auch analytischen Anforderungen gerecht zu werden, verwendet das Datenbanksystem ein dreischichtiges Modell um Daten zu schreiben und zu lesen. Abb. 11 zeigt diesen Lebenszyklus. Ein Datensatz durchläuft nach dem Schreiben in die Datenbank drei Speicher. Der *L1-DELTA* Speicher ist für die Anfrage des neuen Datensatzes ausgelegt. Als relationaler Speicher ist er für eine schnelle Abarbeitung von Insert und Delete Anweisungen optimiert (Bei einem Update wird intern ein neuer Datensatz angelegt [Berg2012]). Die Daten werden schließlich in den *L2-DELTA* Speicher überführt, der sie in Spalten organisiert. Dort finden erste

Komprimierungen mittels Dictionary encoding statt. Im *MAIN STORE* sind die Daten schließlich maximal Komprimiert und leseoptimiert. [SFLCPB2011]

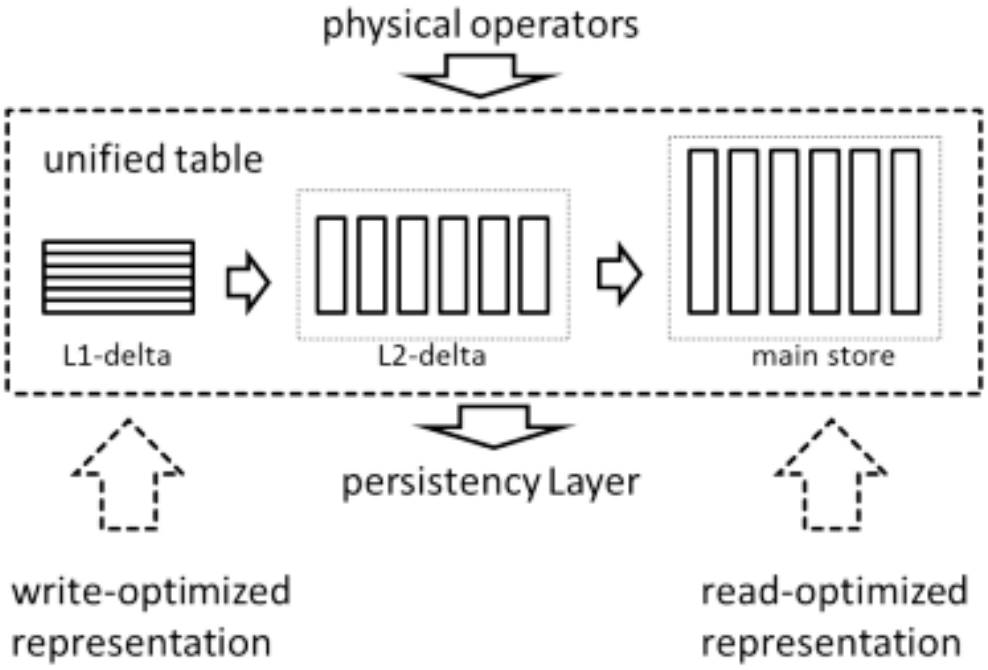


ABB. 11 LEBENSZYKLUSMANAGEMENT VON DATENBANKEINTRÄGEN[SFLCPB2011]

Auf der logischen Ebene hat der Datenbankentwickler, etwa mittel SAP HANA Studio jedoch die Möglichkeit sowohl spalten- als zeilenorientierte Tabellen anzulegen. Dies beeinträchtigt jedoch nicht das zuvor genannte Modell, sondern ist lediglich die logische Repräsentation der Daten. So kann aber der Entwickler selbst entscheiden, ob eine Tabelle lese- oder schreiboptimiert repräsentiert werden sollen.

4.1.2. Datenmodellierung mit SAP HANA Studio

Für die Datenbankentwicklung und Administration kann die auf Eclipse basierende Entwicklungsumgebung SAP HANA Studio verwendet werden. Die Entwicklungsumgebung kann mit einem oder mehreren HANA Instanzen verbunden werden und kann somit auch als Administrationswerkzeug für verschiedene Projekte eingesetzt werden. Für Administration und Datenbankentwicklung können spezifische Perspektiven, die die Elemente von HANA Studio für spezielle Zwecke anordnen, verwendet werden. Es sind folgende standardisierte Perspektiven verfügbar:

- *ADMINISTRATION CONSOLE*
- *MODELER*
- *RCP PERSPECTIVE*

- *RESOURCE*
- *SVN REPOSITORY EXPLORING*
- *TEAM SYNCHRONIZING*

Für die Datenbankentwicklung ist dabei vorzugsweise die *MODELER* Perspektive zu verwenden.

Die *NAVIGATOR* Ansicht zeigt alle Elemente einer HANA Instanz. In Abb. 12 ist der *CATALOG* einer Instanz (hier mit dem Namen *HANA_WIEDER*) zu sehen. Unter dem Schema *ZUGMONITOR* sind im Ordner *TABLES* alle Tabellen des Schemas aufgeführt (Das Beispielszenario aus dem dieser Screenshot stammt wird in den Kapitel 6 und 7 ausführlich beschrieben). An dem Icon vor dem Tabellenname ist auch zu erkennen, ob die Tabelle spalten- oder zeilenbasiert repräsentiert wird.

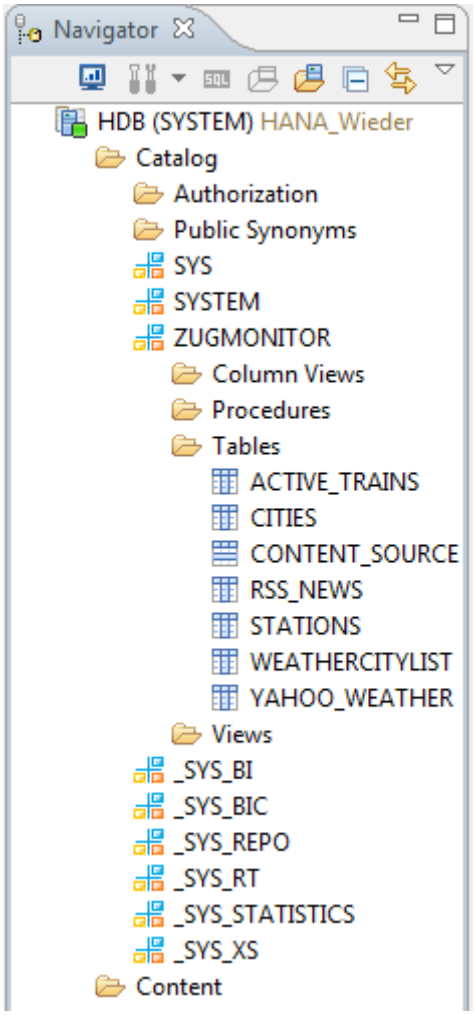


ABB. 12 NAVIGATOR

Zur Erstellung von Datenbankschemata und Tabellen kann ein grafischer Editor verwendet werden (Abb. 13). Hier kann ebenso ausgewählt werden, ob die Tabelle spalten- oder zeilenbasierte repräsentiert werden soll.

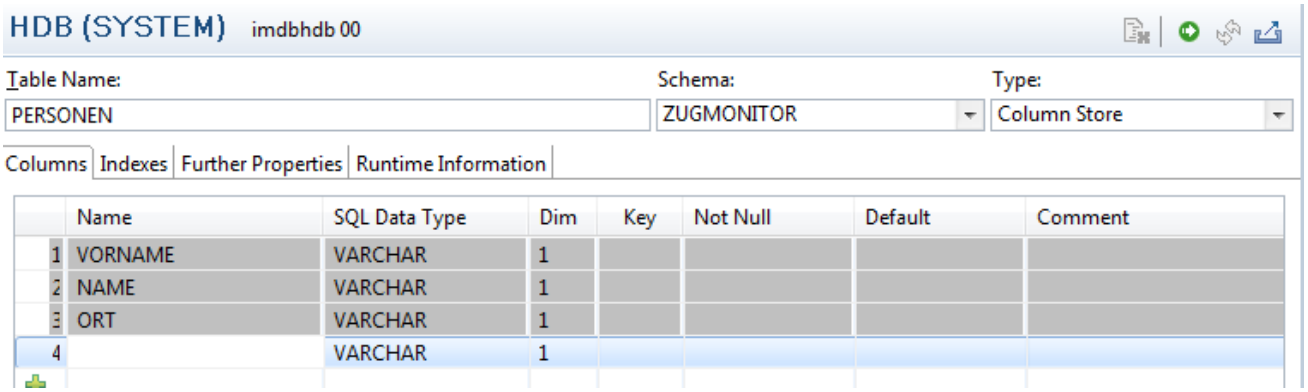


ABB. 13 ERSTELLEN EINER TABELLE MIT DEM GRAFISCHEN EDITOR

Für die Kommunikation mit der Datenbank kann der Entwickler zudem auch einen SQL Editor verwenden, der standardisierte SQL Befehle sowie SAP spezifisches SQLScript interpretieren kann. Zur Generierung von SQL Select Befehlen ist zudem ein Grafischer Editor (Visual SQL) vorhanden mit dem beispielsweise leicht Joins über mehrere Tabellen per *DRAG AND DROP* erstellt werden können. Abb. 14 zeigt eine Join-Anweisung über zwei Tabellen erstellt mit Visual SQL.

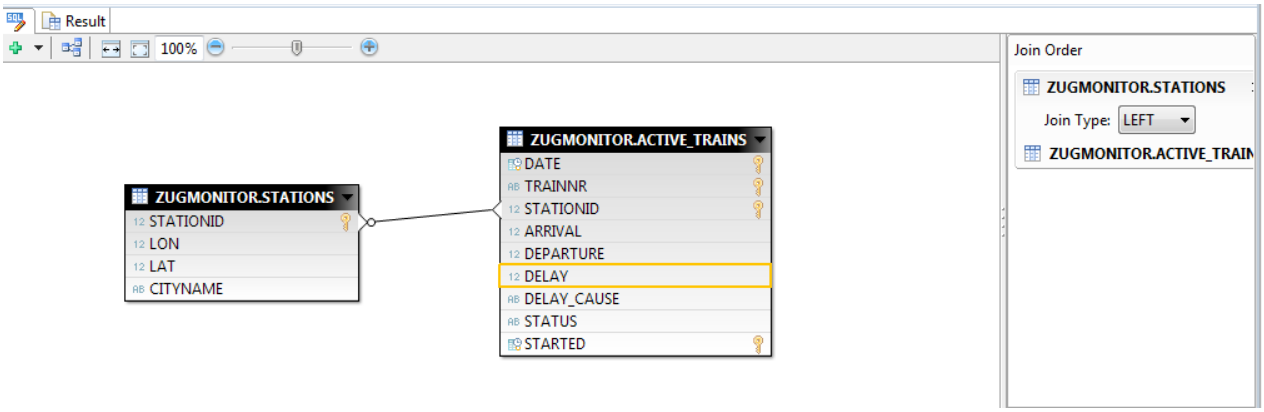


ABB. 14 JOIN ÜBER ZWEI TABELLEN MIT VISUAL SQL

In der *MODELER PERSPECTIVE* wird neben dem *CATALOG* Ordner noch der Ordner *CONTENT* angezeigt (Abb. 15). Er enthält sämtliche modellierte Objekte. Diese sind im allgemeinen *ATTRIBUTE VIEWS*, *ANALYTIC VIEWS* und *CALCULATION VIEWS* welche im Folgenden näher erklärt werden.

Die Objekte können in Pakete gegliedert werden, welche jedoch keine zugriffsrechtlichen Beschränkungen wie etwa in der Programmiersprache Java implementieren. Sie dienen lediglich einer besseren Übersicht.

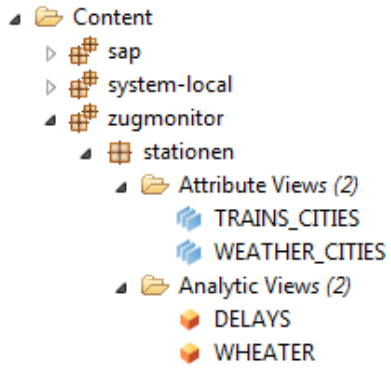


ABB. 15 CONTENT ORDNER IM NAVIGATIONS FENSTER

Um die Daten aus der Datenbank für Business Intelligence Anwendungen aufzubereiten, etwa die Erstellung von Business Intelligence typischen Würfeln, stehen in SAP HANA sogenannte Views zur Verfügung. Es gibt drei verschieden Arten von Views:

Attribute Views

Mit einer *ATTRIBUTE VIEW* ist es möglich Inhalte aus mehreren Tabellen zusammenzuführen. Nach Anlegen einer View können die ausgewählten Tabellen mithilfe eines grafischen Editors per *DRAG AND DROP* verknüpft werden. Die Verknüpfung der Tabellen wird intern mit SQL Joins realisiert, sodass an dieser Stelle die fehlende Fremdschlüsselbeziehung bei der Tabellenerstellung durch referentielle Joins in einer *ATTRIBUTE VIEW* ersetzt werden kann. Für eine View (dies gilt also auch für *ANALYTIC*- und *CALCULATION VIEW*) müssen weiterhin die Output Attribute definiert werden. Diese Attribute werden bei Verwendung der View ausgegeben. Wird ein Attribut als Key-Attribut gesetzt so stellt es den Primärschlüssel der View dar. Eine *ATTRIBUTE VIEW* kann in *ANALYTIC*- und *CALCULATION VIEWS* ähnlich einer Tabelle verwendet werden.

Abb. 16 zeigt die grafische Benutzeroberfläche zur Erstellung einer *ATTRIBUTE VIEW*. Die beiden Tabellen sind dabei über eine referentielle Join Anweisung verknüpft. Im Fenster rechts sind die Attribute des Outputs aufgelistet.

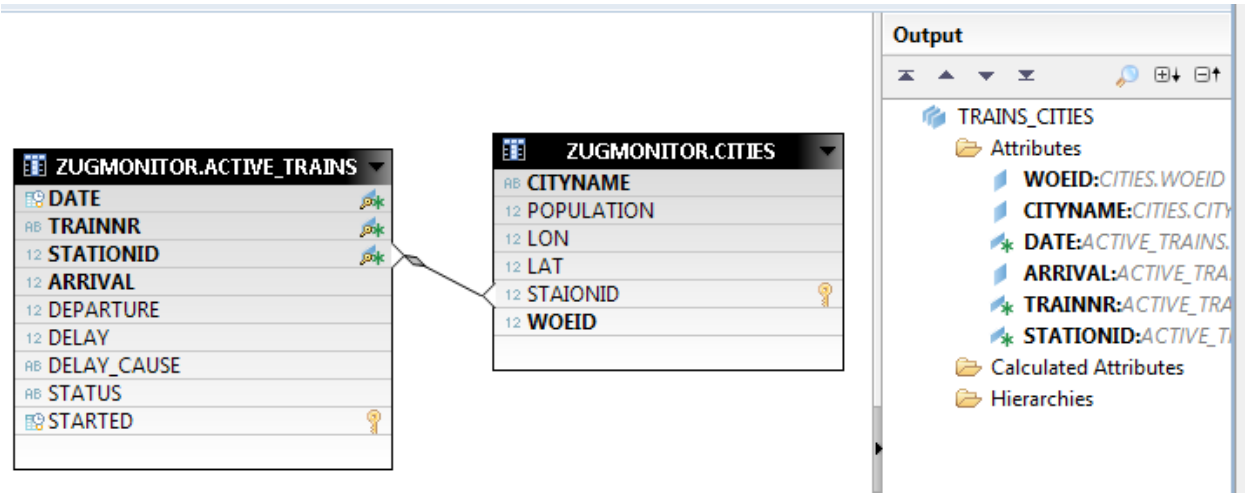


ABB. 16 ATTRIBUTE VIEW - REFERENTIELLER JOIN

Analytic Views

Die *ANALYTIC VIEW* ist die für Business Intelligence Systeme relevanteste View. Diese View stellt im Sinne des Business Intelligence einen Würfel dar.

Eine Analytic View kann folgendes enthalten:

- *„Eine Faktentabelle mit Kennzahlen*
- *Eine beliebige Anzahl von Attribute Views*
- *Eine beliebige Anzahl von anderen Tabellen“*

(übersetzt aus dem Englischen) [Berg2012, S. 237]

Eine *ANALYTIC VIEW* muss jedoch keine *ATTRIBUTE VIEWS* enthalten und kann somit direkt auf den Tabellen des Datenbankschemas angewendet werden. In der *ANALYTIC VIEW* ist der grafische Editor in zwei Bereiche aufgeteilt: *DATA FOUNDATION* und *LOGICAL VIEW*. Die *DATA FOUNDATION* zeigt die physischen Tabellen der *ANALYTIC VIEW* was bedeutet, dass die ausgewählten *ATTRIBUTE VIEWS* hier nicht angezeigt werden. In der *DATA FOUNDATION* Ansicht können Joins über Tabellen angewendet werden und es müssen die in der View zu verwendenden Attribute festgelegt werden. Zusätzlich werden die Kennzahlen der *ANALYTIC VIEW* als *MEASURES* gekennzeichnet. Diese ausgewählten Attribute, werden in der *LOGICAL VIEW* als *DATA FOUNDATION* angezeigt und können dort wiederum mit *ATTRIBUTE VIEWS* verknüpft werden. Die *LOGICAL VIEW* enthält stets eine *DATA FOUNDATION*, unabhängig davon aus wie vielen

Tabellen die Attribute stammen. Auf Abb. 17 ist der grafische Editor einer *ANALYTIC VIEW* zu sehen. Die Kennzahl dieser View ist das Feld *DELAY*.

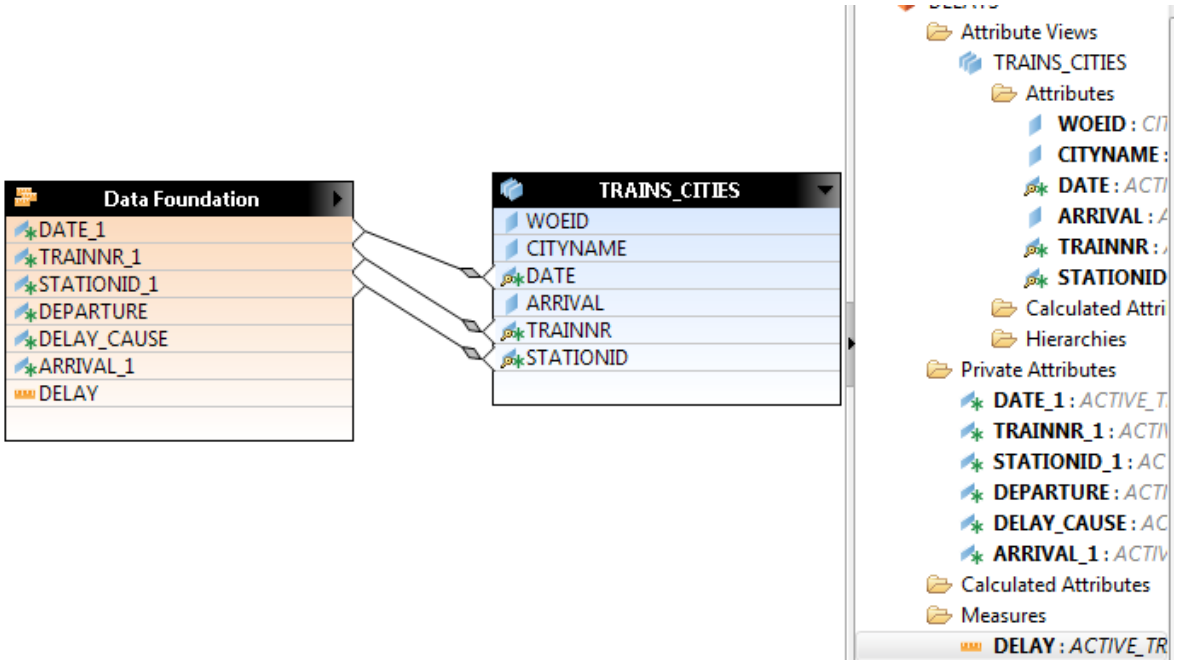


ABB. 17 ANALYTIC VIEW

Calculation Views

Für komplexe Analysen können *CALCULATION VIEWS* verwendet werden. Grundsätzlich können sie für zwei Einsatzgebiete verwendet werden.

- *„Zur Kombination von zwei oder mehreren Analytic Views; zum Beispiel, Plan und ist Daten*
- *Für komplexe Transformationen, mit ein oder zwei Analytic Views.“*

(übersetzt aus dem Englischen) [Berg2012, S. 249]

Eine *CALCULATION VIEW* kann entweder mithilfe des grafischen Editors oder mit SQLScript erstellt werden, wobei sich der grafische Editor von dem der *ATTRIBUTE*- und *ANALYTICAL VIEW* stärker unterscheidet.

Mit dem Grafischen Editor können neben Join-Operationen (Abb. 18) auf Tabellen und anderen Views auch noch zusätzlich die Operationen *UNION*, *PROJECTION* und *AGGREGATION* verwendet werden.

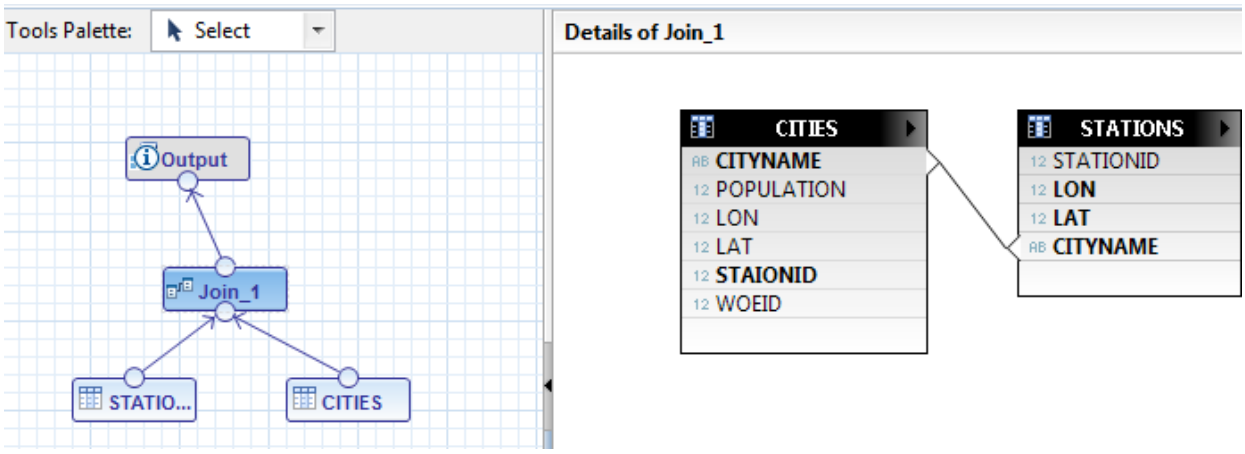


ABB. 18 GRAFISCHER EDITOR FÜR CALCULATION VIEW

Die eigentliche Stärke der *CALCULATION VIEW* ist jedoch die Verwendung der SQLScript Methode zur Erstellung der View. In einem Texteditor wird es dem Entwickler so ermöglicht komplexe Algorithmen zur Berechnung einer *CALCULATION VIEW* mit SQLScript zu erstellen. Die Ausgabe eines Objektes wird dabei über eine Variable (*VAR_OUT*) gesteuert. Beispiele zur Calculation View sind in Abschnitt 7.4.1 erläutert. Den generellen Aufbau zeigt Abb. 19 mit dem SQL Editor in der Mitte, der grafischen Darstellung des Ablaufes links und der Definition für Output und Input Parameter rechts.

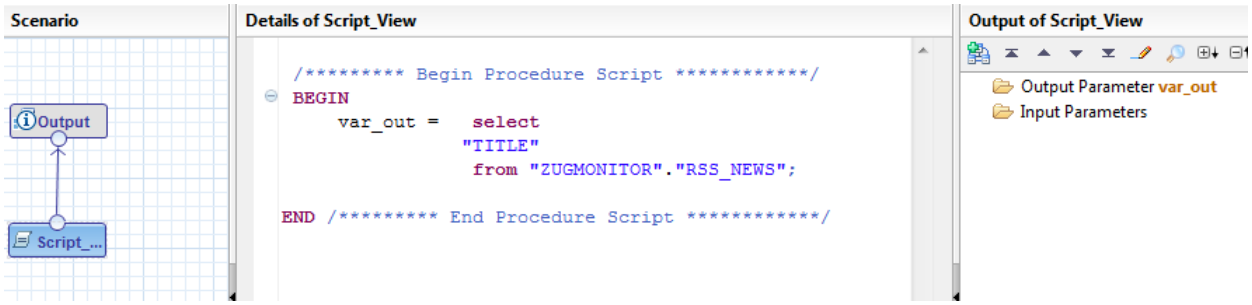


ABB. 19 CALCULATE VIEW

4.2 Verschieden Versionen von SAP HANA

SAP vertreibt SAP HANA in zwei verschiedenen Versionen, welche wiederum in drei unterschiedlichen Ausführungen erworben werden können. Zuerst müssen sich Kunden entscheiden ob sie SAP HANA als „Standalone Version“ oder als Datenbanksystem für SAP NetWeaver BW. Unabhängig davon muss zwischen den drei Softwareausführungen „Enterprise Extended Edition“, „Enterprise Edition“ und „Platform

Edition“ gewählt werden. Alle Versionen von SAP HANA haben dabei die gleichen Hardwareanforderungen. Tabelle 3 zeigt einige beispielhafte Konfigurationen eines IBM Systems (Es werden ebenso Konfigurationen von den Herstellern Dell, Intel und HP angeboten [Berg2012]). Mit den Details der unterschiedlichen Versionen beschäftigen sich die folgenden Abschnitte.

Building Block	**Server (MTM)**	**CPUs**	**Memory**	**Log Storage**	**Data Storage**
XS	X36CP90 X5 (7147-H1x)	2x Intel Xeon E7-2870	128GB DDR3 (8x 16GB)	8x 50GB 1.8“ MLC SSD	8x 300GB 10k SAS HDD
S	X3690 X5 (7147-H2x)	2x Intel Xeon E7-2870	256GB DDR3 (16x 16GB)	8x 50GB 1.8“ MLC SSD	8x 300GB 10k SAS HDD
SSD	X3690 X5 (7147-H3x)	2x Intel Xeon E7-2870	256GB DDR3 (16x 16GB)	10x 200GB 1.8“ MLC SSD (combined log and data)	
S+	X3950 X5 (7143-H1x)	2x Intel Xeon E7-8870	256GB DDR3 (16x 16GB)	320GB FusionIO	8x 600GB 10k SAS HDD
M	X3950 X5 (7143-H2x)	4x Intel Xeon E7-2870	512GB DDR3 (32x 16GB)	640GB FusionIO	8x 600GB 10k SAS HDD
L option (M+ Scalability Kit)	X3950 X5 (7143-H2x + 7143-H3x)	8x Intel Xeon E7-2870	1TB DDR3 (64x 16GB)	2x 640GB FusionIO	16x 600GB 10k SAS HDD

TABELLE 3 HARDWARE KONFIGURATIONSOPTIONEN FÜR SAP HANA NACH[BERG2012]

4.2.1. SAP HANA Standalone Version – SAP HANA 1.0

Die Standalone Version ist unabhängig von weiteren Produkten als Datenbanksystem einsetzbar. Häufig wird diese Version auch als SAP HANA 1.0 oder als Enterprise Version bezeichnet. Das bei AWS gehostete SAP HANA ONE ist ebenfalls hier einzuordnen und unterscheidet sich prinzipiell nur dadurch, dass die Hardware nicht erworben werden muss.

Wenn die Standalone Version eingesetzt werden soll, bedeutet das im Gegensatz zu der mit NetWeaver verbundenen Version mehr Flexibilität bei der Integration mit SAP und nicht-SAP Produkten. Im Gegenzug bedeutet dies auch einen erhöhten Aufwand, was vor allem die Entwicklung von ETL Prozessen und die Anbindung von Business Intelligence Anwendungen betrifft. Abschnitt 4.4 geht auf diese Problemstellung ein. [Berg2012]

4.2.2. SAP HANA for SAP NetWeaver BW – SAP HANA 2.0

SAP HANA for SAP NetWeaver BW (auch SAP HANA 2.0) ermöglicht es SAP Kunden ihr Business Intelligence System deutlich zu beschleunigen. Hier ersetzt SAP HANA das Datenbanksystem (Oracle oder DB2) von SAP NetWeaver BW. Es ist sowohl möglich ein bestehendes SAP NetWeaver BW System zu erweitern, als auch ein komplett neues mit SAP HANA zu entwickeln. Dadurch dass SAP HANA nun direkt an SAP NetWeaver BW gekoppelt ist, können die eventuell bereits vorhanden Business Intelligence Prozesse (wie z. B. ETL Prozesse) verwendet werden. Allerdings kann SAP HANA 2.0 nur mit bestimmten Versionen verwendet werden. Derzeit wird SAP NetWeaver BW 7.3 (Unicode) und SAP BusinessObjects 4.0 (falls SAP BusinessObjects verwendet werden soll) vorausgesetzt. Um die SAP HANA Datenbank vollständig zu migrieren sind jedoch noch einige Anpassungen nötig. So wird zum Beispiel empfohlen aggregierte Tabellen aus dem Datenbankschema zu entfernen, da sie mittels HANA zur Laufzeit berechnet werden können. [Berg2012]

4.2.3. SAP HANA Softwareausführungen

Die drei Softwareausführungen „Enterprise Extended Edition“, „Enterprise Edition“ und „Platform Edition“ sind unabhängig von der gewählten SAP HANA Version (Standalone oder mit SAP NetWeaver BW) und unterscheiden sich in den enthaltenen Softwareerweiterungen.

Alle Softwareausführungen enthalten SAP HANA Studio, SAP HANA Information Composer, SAP HANA Client, SAP HANA Client for Excel, SAP HANA User Interface for Information Access(INA), SAP HANA Database, SAP HANA Host Agent und Diagnostics Agent.

Platform Edition

Die Platform Edition ist für nicht-SAP Umgebungen geeignet. Sämtliche Prozesse für ETL und Business Intelligence müssen selbst implementiert werden, falls entsprechende Anwendungen wie SAP Data Services (Abschnitt 4.4.1) nicht bereits Lizensiert sind oder hinzu gekauft werden sollen. [Berg2012]

Enterprise Edition

Das Paket enthält SAP Data Services (Abschnitt 4.4.1) um Daten aus beliebigen Datenquellen in das SAP HANA System zu laden, sowie die weiteren in Abschnitt 4.4.1 beschriebenen ETL Tools SAP Landscape Transformation Tool (SLT), Transformation Replication Server. Zusätzlich ist SAP HANA Direct Extractor Connection (DXC) enthalten. Diese Variante wird für den Einsatz in Unternehmen empfohlen die eine Trigger basierte Replikation der Daten benötigen. [Berg2012]

Enterprise Extended Edition

Für Unternehmen die bereits SAP ERP oder Business Intelligence Anwendungen von SAP verwenden wird die Enterprise Extended Edition empfohlen. Mit dem enthaltenen Sybase Replication Server, der ebenfalls in Kapitel 4.4.1 beschrieben wird, können die Tabellen aus dem SAP ERP System in Echtzeit in SAP HANA gespiegelt werden um daraus wiederum mittels der SAP HANA Appliance möglichst schnell Business Intelligence Anfragen auszuwerten. [Berg2012]

4.3 Datensicherheit bei Stromausfall

Um den Anforderungen eines vollwertigen Datenbanksystems zu genügen muss auch SAP HANA die Anforderungen des ACID Prinzips (Atomicity, consistency, isolation und durability) erfüllen. Vor allem das Prinzip der durability, also der dauerhaften Speicherung von Daten ist durch ein Datenbanksystem, dessen Daten vollständig im Hauptspeicher liegen nicht gegeben. Daher verwendet auch SAP HANA klassische Festplatten um Daten dauerhaft zu speichern. Dazu werden die Datenbankseiten (pages), sobald sie durch eine Transaktion manipuliert wurden markiert und in Intervallen auf Festplatten geschrieben. Zusätzlich werden alle Änderungen die durch Transaktionen erfolgen in einem Datenbanklog gesichert.

Die geänderten Datenbankseiten werden als Sicherungspunkte alle fünf Minuten (default Einstellung) auf Festplatten geschrieben. Der Datenbanklog wird dagegen Synchron mit den Transaktionen geschrieben, was zur Folge hat, dass eine Transaktion fehlschlägt, wenn nicht auch der Datenbanklog geschrieben wurde. Nach einem Stromausfall kann SAP HANA aus den Sicherungspunkten wiederhergestellt werden. Die restlichen in der Zeit seit der letzten Sicherung verarbeiteten Transaktionen werden mittels des Datenbanklogs wiederhergestellt. [Berg2012]

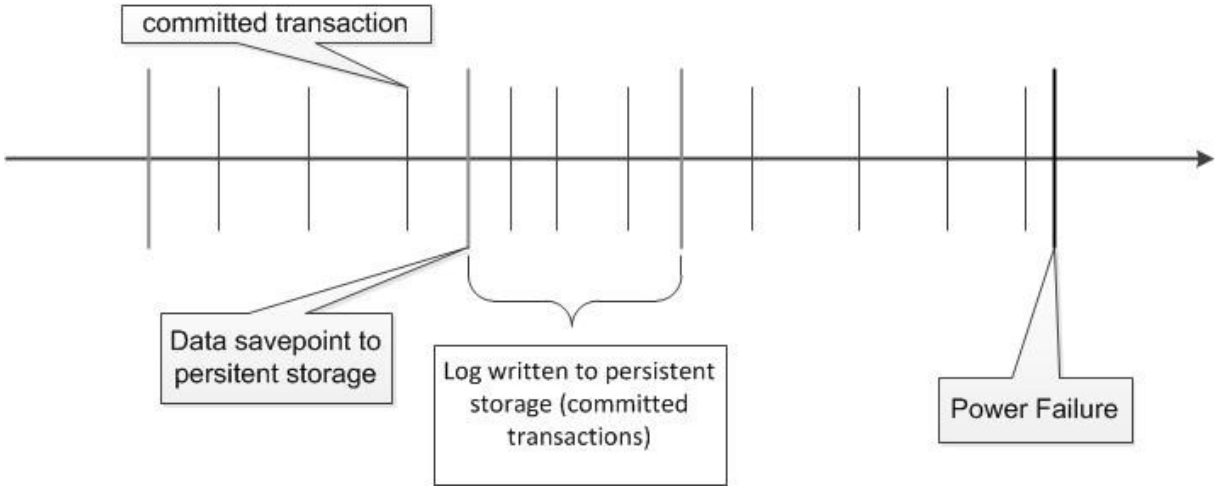


ABB. 20 WIEDERHERSTELLUNG NACH STROMAUSFALL. NACH [BERG2012]

4.4 SAP HANA für Business Intelligence

SAP HANA ist besonders für Business Intelligence Anwendungen geeignet und wird auch in hier im Weiteren für diesen Zweck betrachtet. Für die Integration in den Unternehmen bietet die SAP mehrere Möglichkeiten. Dabei steht in den aktuellen Versionen von SAP HANA vor allem die Anbindung an weitere SAP Produkte im Vordergrund. Abb. 21 zeigt eine Übersicht der SAP HANA Appliance (SAP HANA 1.0) und Anwendungen zu denen SAP HANA Schnittstellen verfügt.

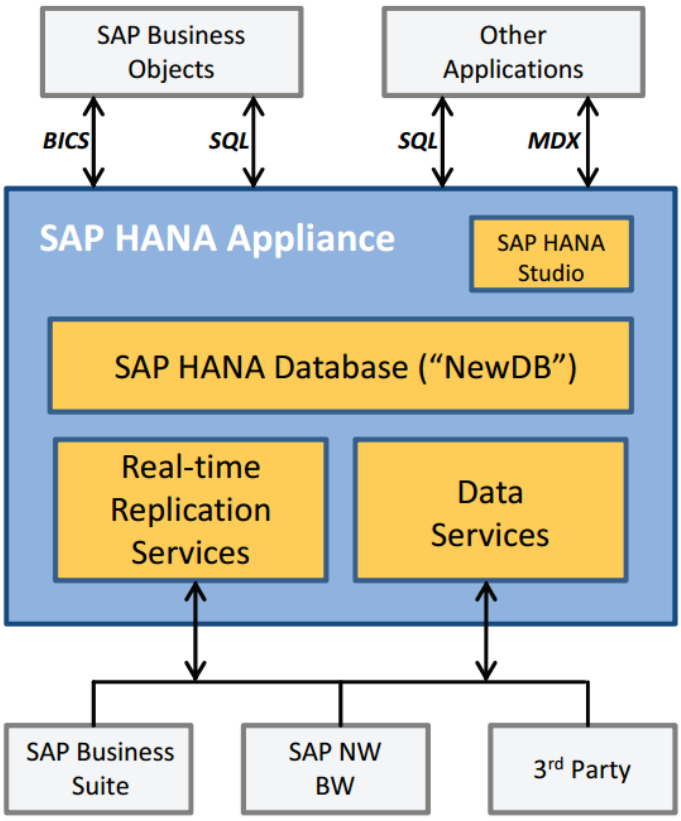


ABB. 21 SAP HANA IM BUSINESS INTELLIGENCE UMFELD [FÄRBER2011]

4.4.1. ETL Tools

Um Daten in die SAP HANA Datenbank zu laden stehen mehrere Methoden zur Verfügung. Neben den in Abb. 21 gezeigten Anwendungen ist ein Zugriff über JDBC/ODBC möglich. Daten können so direkt mittels SQL Anweisungen an die Datenbank gesendet werden. Aufgrund der Tatsache, dass die im folgenden vorgestellten ETL Tools zusätzliche SAP Produkte nicht in der SAP HANA ONE Lizenz enthalten sind, wurde in dieser Arbeit der Zugriff über JDBC realisiert.

Die Produkte SAP Business Suite und SAP NW BW sowie einige Produkte von Drittanbietern können folgende Methoden nutzen.

SAP Data Services

Das ETL Tool SAP Data Services bietet im Vergleich zu den weiteren hier vorgestellten Tools die größte Flexibilität. Es können dabei Daten aus SAP sowie nicht SAP Systemen nach SAP HANA übertragen werden. Darüber hinaus können die Daten auch umfangreich transformiert werden bevor sie an die SAP HANA Datenbank übertragen werden. Dagegen ist aber keine Echtzeitreplikation der Daten möglich.

SAP Landscape Transformation (SLT)

Die hauptsächlich für SAP Quellsysteme eingesetzte „Data Provisioning“ *(“SLT/trigger-based replication is the primary data provisioning mechanism for SAP source Systems [...].”* [Berg2012 S. 326]) Methode SLT benötigt im Gegensatz zu SAP Data Services einen Replikations Server (SLT Replication Server). SLT erkennt Änderungen in den Quellsystemen und so den Datenbestand nahezu in Echtzeit an die SAP HANA Datenbank weitergeben. Die Spiegelung des Datenbestandes kann aber auch periodisch erfolgen. Die Transformation der Daten sowie die Verwendung von SLT mit nicht SAP Systemen ist nur begrenzt möglich. SLT kann in SAP HANA Studio administriert werden.

Sybase Replication Server

Mit dieser Methode ist es möglich die Daten der Quellsysteme in Echtzeit, also gleichzeitig mit einer Änderung im Quellsystem, auf das SAP HANA System zu spiegeln. Sybase Replication Server umgeht dafür die Anwendungsschicht des Quellsystems und verwendet Log Tabellen der Quelldatenbank um Änderungen zu erkennen. Daher ist aber auch keine Transformation der Daten möglich und diese Methode funktioniert nur für IBM DB2 für LUW Datenbanksysteme. Sybase Replication Server kann in SAP HANA Studio mit der Load Controller Komponente administriert werden.

4.4.2. Business Intelligence Tools

SAP HANA ist wie schon erläutert keine Business Intelligence Anwendung. Es können zwar mittels *ANALYTIC VIEWS* (vgl. Kapitel 4.1.2) OLAP-Würfel angelegt und auch in SAP HANA Studio begrenzt ausgewertet werden, was aber nur zu Testzwecken geeignet ist. Um weiter Analysen und Reportings durchzuführen wird daher ein

Business Intelligence Tool benötigt, das diesen Aufgabenbereich abdeckt. Wie in [Berg2012] erläutert sind lediglich zwei Produkte, nämlich SAP BusinessObjects und Microsoft Excel für die Verwendung mit SAP HANA zertifiziert. SAP teste aber derzeit weitere nicht SAP Business Intelligence Tools. Als offene Schnittstellen stehen auch hier ODBC und JDBC sowie die Anfragesprachen SQL und MDX zur Verfügung. In dieser Arbeit wird daher Microsoft Excel verwendet.

5. Bedeutung von SAP HANA für Business Intelligence Systeme

Business Intelligence Systeme orientieren sich seit Jahren, an den in Kapitel 2 vorgestellten Konzepten. Gerade die Referenzarchitektur für das Data Warehouse (vgl. Abschnitt 2.4) ist in Unternehmen etabliert. Wie auch in Abschnitt 2.5.3 beschrieben, kann diese Referenzarchitektur, aufgrund der heutigen technischen Entwicklung durchaus infrage gestellt werden. SAP versucht mit SAP HANA einige dieser in Abschnitt 2.5.3 beschriebenen Probleme und Anforderungen zu bewältigen. Die folgenden Abschnitte greifen die Kritikpunkte aus Abschnitt 2.5.3 auf und erläutern inwiefern SAP HANA für diese Anforderungen geeignet ist.

5.1 Zusammenführung von OLTP und OLAP

Die Trennung von OLTP und OLAP Daten war in den frühen 90er Jahren aufgrund der benötigten Rechenleistung durch aufwendigere Analyseaufgaben der Unternehmen ein notwendiger Schritt [PlZe2011]. Erst in den vergangenen Jahren entwickelten sich die verfügbaren Hardwaresysteme so weit, dass es möglich scheint analytische Problemstellungen wieder auf operativen Daten auszuführen ohne die transaktionalen Prozesse zu beeinflussen.

5.1.1. In-Memory statt Festplatten

Die Rechengeschwindigkeit von Prozessoren ist seit den 90er Jahren stark angestiegen. Die lese- und Schreibgeschwindigkeiten von Festplatten hat sich dabei kaum beschleunigt. Der Zugriff auf eine Festplatte von einer Anwendung bedeutet daher immer auch Performanz Nachteile. Würde ein Business Intelligence System direkt auf eine Datenbank eines operativen ERP Systems zugreifen, hätten die Anwender des ERP Systems mit längerer Latenz zu rechnen.

Mit der In-Memory Technologie in SAP HANA ist die Festplatte als Hauptspeichermedium aus dem Datenbanksystem entfernt worden. Dadurch, dass alle Daten im Hauptspeicher verfügbar sind, sind schon deshalb etwa 120fach schnellere Lese- und Schreibvorgänge zu erwarten [PlZe2011].

5.1.2. Zeilen- und Spaltenorientiertes Datenbanksysteme

Transaktionale Systeme benötigen relationale (zeilenorientiertes) Datenbanksysteme. Analytische Systeme benötigen Spaltenorientierte (oder spezielle Multidimensionale) Datenbanksysteme.

Dieser Ansatz wird in der Literatur (z. B. [BaGü2009]) und in produktiven Business Intelligence Systemen angewendet. Auch hier wird in Kapitel 2 auf diesen Ansatz eingegangen. PlZe stellt diesen Ansatz, wie ebenfalls in Kapitel 2 beschrieben infrage, denn die Anforderungen an ein Datenbanksystem für analytische und transaktionale Aufgaben unterschieden sich nur marginal (vgl. Abschnitt 2.5.3). Demnach sollten für beide Aufgabenbereiche die Spaltenorientierte Datenbanksystem verwendet werden.

Das Datenbanksystem von SAP HANA geht dabei einen Kompromiss ein. Es bietet dem Entwickler beim Erstellen von Tabellen die Wahl zwischen spalten- oder zeilenorientierten Tabellen. So kann bei der Zusammenführung von analytischen und transaktionalen Systemen explizit unterschieden werden, wie Tabellen verarbeitet werden sollen (Tatsächlich werden Daten in der finalen Persistenz Schicht immer spaltenorientiert im Hauptspeicher abgelegt (vgl. Abschnitt 4.1)).

5.1.3. Aufwendige Berechnungen auf Datenbankebene

Neben der In-Memory Technologie ist ein SAP HANA System auch mit enormer Rechenleistung ausgestattet. Diese Rechenleistung wird nicht ausschließlich zur Haltung der Daten im Hauptspeicher benötigt, sondern kann dazu verwendet werden, aufwendige Berechnung schon auf Datenbankebene durchzuführen. Mit SQLSrcipt, einer proprietären Erweiterung des SQL Standards von SAP (vgl. 7.3.1), steht dem Entwickler ein umfangreicherer Sprachumfang zur Verfügung, der Komplexe Berechnungen ermöglicht. Eine Anwendung, die zur Berechnung einer Kennzahl bisher eine große Menge von Daten laden musste um diese dann auszuwerten, müsste bei einer Datenbankseitigen Implementierung der Kennzahlberechnung lediglich das Ergebnis der Berechnung laden. Gerade mobile Geräte oder Business Intelligence Anwendungen in Webbrowsern könnten davon profitieren.

5.1.4. Bedeutung für das Data Warehouse

Wird das SAP HANA Datenbanksystem für OLAP Anwendungen als Data Warehouse eingesetzt, sind den Entwicklern auf Basis der zuvor vorgestellten abgrenzenden Eigenschaften des Datenbanksystems neue Möglichkeiten gegeben. Ein bedeutender Vorteil der durch die Nutzung der Spalten- und In-Memory Architektur genutzt werden kann ist die Berechnung von Aggregaten bei Bedarf, anstatt sie in speziellen Tabellen zu speichern. *„Da die Erstellung von Aggregaten dem Lesen aller Inhalte einer Spalte entspricht, hat die Anzahl der Datensätze nur wenig Auswirkung auf die Antwortzeit. In einem zeilenbasierten Speicher wächst die Antwortzeit dagegen linear mit der Anzahl der Datensätze“* [Plattner2012 S. 3]. Dadurch kann nicht nur das physische Datenbankschema begrenzt werden, sondern auch die Flexibilität von analytischen Anfragen erhöht werden. Bei der Aggregation in der Zeitdimension kann so beispielsweise für jede Anfrage neu entschieden werden ob eine Aggregation nur auf Wochenbasis oder gar auf Stundenbasis erfolgen soll. Sollte darüber hinaus SAP HANA als einziges Datenbanksystem für OLTP und OLAP eingesetzt werden (vgl. Abschnitt 5.3.2), so entfällt in diesem Fall das Data-Warehouse. Dies hat jedoch zur Folge, dass das Datenbankschema für dieses System sowohl für OLAP als auch für OLTP optimiert sein sollte.

5.2 Einsatzmöglichkeiten von SAP HANA außerhalb der „SAP Welt“

Auch wenn SAP HANA wie in Kapitel 4 beschrieben zunächst als reines Datenbanksystem betrachtet wird, so wird bei genauerer Untersuchung bald auch klar, dass die Appliance seine Stärken vor allem in Kombination mit weiteren SAP Produkten ausspielen kann. Deutlich wird dies gerade im Bereich des Business Intelligence, wo große Datenmengen verarbeitet werden müssen. Business Intelligence mit Echtzeitcharakter kann SAP HANA somit nicht alleine, sondern nur mit weiteren SAP Produkten leisten. Dies muss allerdings nicht zwingend in einer restriktiven Produktpolitik begründet sein, wenn man bedenkt, dass SAP HANA noch ein relativ junges Produkt ist. Im Gegenteil sei SAP derzeit (Stand: Ende 2012) beispielsweise dabei weitere nicht-SAP Business Intelligence Tools für SAP HANA zu Zertifizieren [Berg2012]. Es kann also damit gerechnet werden, dass SAP HANA zukünftig breiter für SAP unabhängige Produkte offen steht, auch wenn der Einsatz in einem SAP

Umfeld weiterhin vorteilhafter sein wird. Im Folgenden werden Ansätze für die Verwendung von SAP HANA gezeigt, die unabhängig von weiteren SAP Produkten sind.

5.2.1. SAP HANA als Hauptpersistenz Schicht für Unternehmensanwendungen

Als ein voll funktionsfähiges Datenbanksystem kann SAP Hana prinzipiell ein relationales Datenbanksystem für operative Anwendungen wie einem ERP System ersetzen, was nicht zuletzt durch die Verwendung von zeilenorientierten Tabellen ermöglicht wird. Schlüsselthemen wie SQL Konformität, Rechteverwaltung, Transaktions Management, sowie die gesicherte Speicherung der Daten sind in SAP HANA vorhanden. Dies soll jedoch nicht implizieren, dass das Datenbanksystem einer bestehenden Anwendung ohne großen Aufwand mit einem SAP HANA System ersetzt werden könnte.

Für diesen Zweck alleine werden die Möglichkeiten von SAP HANA jedoch zu wenig genutzt, was den Einsatz nur schwer rechtfertigen könnte.

5.2.2. SAP HANA als Data Warehouse

Auch wenn wie zuvor beschrieben die Nutzung von SAP HANA im Business Intelligence Umfeld ohne Zukauf von weiteren SAP Produkten derzeit erschwert ist, so ist der Einsatz als Datawarehouse nicht vollständig auszuschließen. Denn gerade die schnellen Zugriffszeiten auf Daten und die Möglichkeit der Berechnung von komplexen Analysen auf Datenbankebene machen SAP HANA dennoch interessant für diesen Anwendungszweck.

Obwohl für den ETL Prozess eines der in Kapitel 4.4.1 erwähnten Tools derzeit am besten geeignet ist, so kann bei entsprechendem Entwicklungsaufwand und dem Verzicht auf die Echtzeitreplikation der Quelldaten (was ohnehin nur mit dem Sybase Replication Server möglich ist), das Data Warehouse gefüllt werden.

Als Business Intelligence Tool ist neben den eigenen Produkten nur Microsoft Excel zertifiziert. Dennoch sind weiter BI-Tools nicht explizit von der Verwendung mit SAP HANA ausgeschlossen, wie auch in diesem Kapitel schon beschrieben wurde.

Gerade für den Einsatz mit mobilen Endgeräten könnte diese Einsatzmöglichkeit Interessant sein, wenn die datenbankinternen Möglichkeiten zur Berechnung von Kennzahlen genutzt werden um damit die Ressourcen bei den Endgeräten zu schonen.

5.2.3. Eine Zusammenführung aus operativer Datenbank und Data Warehouse

Der Ansatz die Trennung von OLTP und OLAP typischen Aufgaben aufzuheben und beides wieder in einem Datenbanksystem zu einigen wurde bereits mehrfach erwähnt und wird auch im Zusammenhang mit SAP HANA in Abschnitt 5.3 nochmals aufgegriffen.

Um diesen Abschnitt abzuschließen sei jedoch erwähnt, dass die Zusammenführung der operativen und analytischen Daten mit SAP HANA den in Abschnitt 5.2.2 als umständlich identifizierten ETL Prozess (wenn keine weitere SAP Produkte verwendet werden) ersparen würde.

5.3 Wie SAP HANA Business Intelligence verändern kann

Der Einsatz von SAP HANA im Business Intelligence Umfeld ist vielseitig denkbar: SAP HANA kann als klassisches Data Warehouse eingesetzt werden um dieses zu beschleunigen, oder als dessen Erweiterung eingesetzt werden um komplexe Business Intelligence typische Aufgaben zu bewältigen, welche bisher aus technischer Sicht nicht realisierbar waren. Eine echte Bereicherung des Business Intelligence könnte SAP HANA jedoch leisten, wenn sowohl OLTP als auch OLAP Anwendungen auf einem Datenbanksystem arbeiteten.

5.3.1. SAP HANA als Data Warehouse im SAP Umfeld

Wenn das SAP HANA Datenbanksystem als Data Warehouse eingesetzt werden soll, dann muss es wie jedes andere Datenbanksystem in das System integriert werden. Das gilt sowohl für die ETL Prozesse als auch für Business Intelligence Tools und für das Reporting. Im SAP Umfeld ist diese Lösung derzeit deutlich einfacher zu realisieren, als in Abschnitt 5.2.2 für nicht SAP Systeme beschrieben. Auf Basis eines SAP ERP Systems ist Beispielsweise eine Echtzeitspiegelung der operativen Daten in das SAP HANA System möglich (vgl. Abschnitt 4.4.1 – Sybase Replication Server). Um SAP HANA als Data Warehouse optimal zu nutzen, kann es von Vorteil sein auf die Voraggregation von Tabellen zu verzichten (vgl. Abschnitt 5.1.4).

Weiterhin können durch die Nutzung von Calculation Views (vgl. Abschnitt 4.1.2) aufwendige Berechnungen auf Datenbankeben durchgeführt werden. Als Business Intelligence Tools sind SAP BusinessObjects BI Tool und Microsoft Excel von SAP zertifiziert, welche über ODBC, JDBC und Universe Designer mit dem Information Design Tool (nur SAP Business Objects BI 4.0) mit SAP HANA kommunizieren.

5.3.2. SAP HANA als gemeinsame Persistenz für OLTP und OLAP

Hasso Plattner, Mitbegründer der SAP, setzt sich für die Zusammenführung von OLTP und OLAP Systemen in einem einzigen Datenbanksystem ein: *„I always believed the introduction of so-called data warehouses was a compromise. The exibility and speed we gained had to be paid for with the additional management of extracting, and loading data, as well as controlling the redundancy. For many years, the discussion seemed to be closed and enterprise data was split into OLTP and OLAP“* [Plattner2012 S. 1]. Der offensichtlichste Vorteil der Zusammenführung von OLTP und OLAP ist das Einsparen einer Architektur mit vielen Ebenen und Komponenten. Die Leistungsfähigkeit des SAP HANA Datenbanksystems lässt zudem vermuten, dass komplexe Reportings in Echtzeit aus den aktuellen Daten angefragt werden können, ohne den operativen Betrieb zu stören. Entscheidungen aufgrund von betriebswirtschaftlichen Kennzahlen könnten so früher gefällt werden. Durch die geringere Anzahl Komponenten verringert sich weiterhin der Administrationsaufwand des Systems.

Abschnitt 2.5.3. behandelt bereits Anforderungen und Probleme, welche zeigen dass das Zusammenführen der Welten nicht einfach umzusetzen sein wird. Die verschiedenen Datenbanksysteme in den operativen Systemen in Unternehmen sind im Allgemeinen viel zu heterogen um diese ohne die Transformation und Konsolidierung in einem Data Warehouse zu analysieren.

Für Teilbereiche eines Unternehmens, wie etwa dem ERP System könnte jedoch der Einsatz von SAP HANA als einziges Datenbanksystem für das ERP System und dem Business Intelligence dienen. Für den operativen Betrieb mit zeitkritischen Anforderungen für einzelne Insert oder Update Vorgänge werden in der Architektur weiterhin relationale Tabellen verwendet. Tabellen, die Analysezwecken dienen sollen, werden als spaltenorientierte Tabellen angelegt. Um die Analyse von historischen Daten zu ermöglichen, muss darauf geachtet werden, dass die kompletten historischen Daten eben auch im operativen System bleiben.

Die Konsequenz daraus ist, dass bei der Architektur des Datenbankschemas auf SQL Update Anweisungen verzichtet wird und stattdessen nur mit SQL Insert Anweisungen gearbeitet wird. Dies hat neben der Historisierung auch Vorteile in der Datenkompression und Datenbank Locks. [Plattner2012]

6. Anwendungsszenario

Um die Entwicklung von Anwendungssystemen mit der SAP HANA Appliance zu erproben, wurde zunächst ein Anwendungsszenario ausgewählt. Dieses Szenario sollte thematisch im Bereich Business Intelligence liegen und ferner mit einer größeren Menge an Daten arbeiten.

6.1 Beschreibung des gewählten Szenarios

Es bot sich an frei verfügbare Datenquellen zu verwenden, welche mit einer hohen Frequenz aktualisiert werden. Ferner wurde darauf geachtet, dass die Qualität der Quelldaten gleichbleibend sind um die ETL-Prozesse möglichst effizient zu implementieren.

6.1.1. Übersicht des Anwendungsszenarios

Im gewählten Szenario sollen die Fahrten der Fernverkehrszüge in Deutschland analysiert werden. Hierzu wird eine offene[1] API von „Opendatacity“ [Opendatacity2012] verwendet. Ebenfalls werden Internetmeldungen von ausgewählten Zeitungen und Wetterdaten aus allen deutschen Regionen in den Datenbestand der Anwendung integriert. Hinzu kommen Stammdateninformationen über Städte und Regionen in denen sich Bahnhöfe befinden. Mittels SAP HANA sollen diese Daten mit Blick auf den Zugverkehr ausgewertet werden.
Das Anwendungsszenario basiert im Wesentlichen auf die oben beschriebenen drei Datenquellen, welche nachstehend kurz beschrieben werden.

Fahrtdaten der deutschen Fernverkehrszüge

Die Fahrtdaten des Fernverkehrs werden mittels der *ZUGMONITOR* API von „Opendatacity“ heruntergeladen. „Opendatacity“ selbst ermittelt die Daten durch scraping von öffentlich erreichbaren Fahrplandaten der Deutschen Bahn. Die Daten werden auf Servern der „Süddeutschen Zeitung“ gespeichert und liegen als JSON Dateien seit Oktober 2011 vor. Die Dateien können Tagesbezogen aufgerufen werden

[1] Es wird nur ein Recht auf nicht-kommerzielle Nutzung der Daten eingeräumt (also z. B. bitte keine kostenpflichtige Android-App damit umsetzen). Die Veröffentlichung dieser Informationen als wirkliches Open Data ist Sache der Deutsche Bahn. [Opendatacity]

und enthalten alle Züge dieses Tages. Für jeden Bahnhof sind die Ankunfts- und Abfahrtszeiten sowie gegebenenfalls die Verspätungszeit angegeben.

Nachrichtenmeldungen von deutschen Zeitungen

Mittels einer in Java implementierten Anwendung werden RSS Feeds von Nachrichtenseiten im Internet (u. A.: Spiegel Online, Focus und Tagesschau) geparst und jede Nachricht mit Datum, Titel, Beschreibung und Internetlink in eine Datenbank überführt. Da die SAP HANA Appliance aus Kostengründen nicht dauerhaft aktiv sein kann, werden die Daten in einer MySQL Datenbank auf einem separaten Server gespeichert und in unregelmäßigen Abständen in die SAP HANA Datenbank geladen. Es findet jedoch keine Aggregation oder Transformation der Daten statt, sodass bei gleichem Datenmodell die Daten auch direkt in die SAP HANA Datenbank geladen werden könnten.

Wetterdaten von ausgewählten deutschen Städten

Die Wetterdaten werden ähnlich wie die Nachrichten aus einer RSS Datei gelesen. Als wird hierzu der Yahoo!Weather RSS Feed verwendet. Es kann für jede verfügbare Stadt ein RSS Feed abgerufen werden. Dazu wird in der URL eine eindeutige Kennung übergeben (WOEID = Where on Earth Identifier). Der Feed stellt neben Wetterdaten wie Temperatur, Windgeschwindigkeit und einer textuellen Beschreibung der Wetterbedingungen auch geographische Informationen wie Längen-, Breitengrat, und das Bundesland bereit. Die Wetterdaten werden analog zu den Nachrichten in einer MySQL Datenbank zwischengespeichert.

6.1.2. Architektur

Die Architektur des Anwendungsszenarios besteht aus drei Komponenten(vgl. Abb. 22). Der Kern des Systems bildet die SAP HANA Appliance und insbesondere deren Datenbanksystem. Eine weitere Persistenz Schicht stellt die MySQL Datenbank dar, welche zusammen mit der Serverseitigen Java Anwendung *RSS_READER* als Teil des ETL-Prozesses gesehen werden kann. Der zweite Teil des ETL-Prozesses besteht aus einer weiteren Java Anwendung (*LOAD RSS/WEATHER*), die Daten aus der MySQL Datenbank in die SAP HANA Datenbank lädt. Die Anwendung *ZUG_READER* lädt Anwendergesteuert Zugdaten über die *ZUGMONITOR*-API direkt in die SAP HANA Datenbank. Eine grafische Übersicht der Architektur ist nachstehend abgebildet. Als oberste Schicht übernimmt ein Excel Dokument das Reporting. Dazu werden die in der SAP HANA Datenbank generierten OLAP-Würfel importiert.

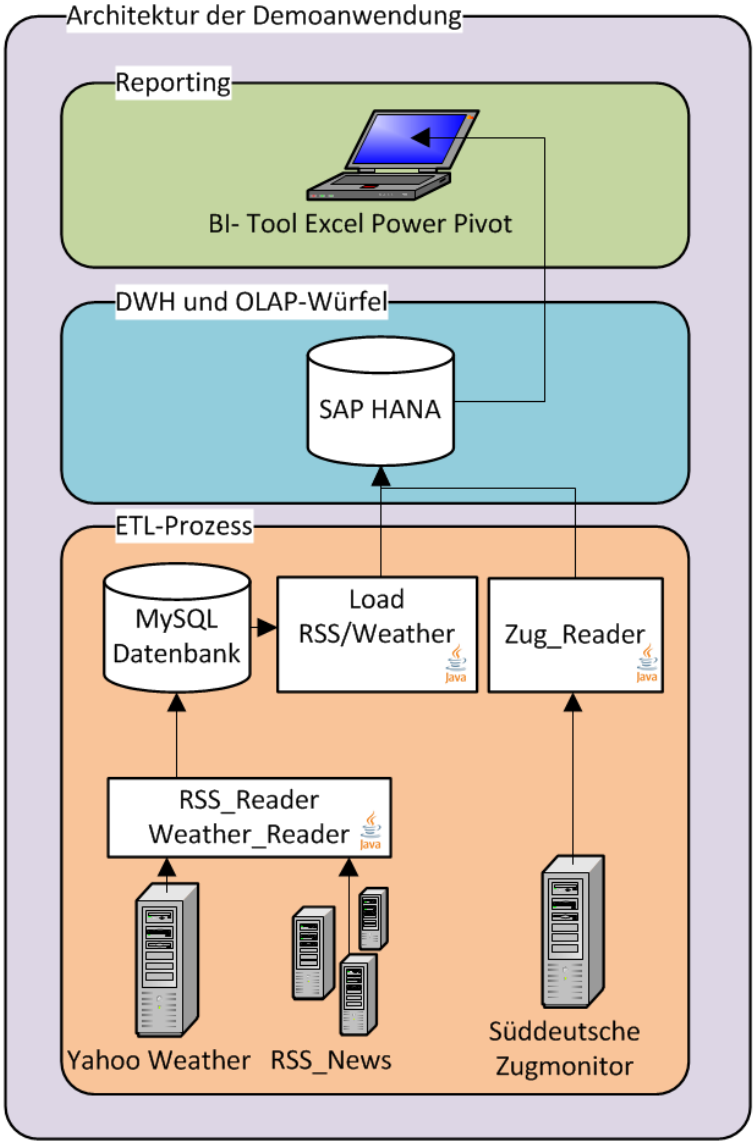


ABB. 22 KOMPONENTENARCHITEKTUR DER DEMOANWENDUNG

SAP HANA Komponente

Die SAP HANA Komponente ist die zentrale Komponente für die Demoanwendung. Alle Daten aus den Quellsystemen werden entweder direkt (*ZUG_READER*) oder indirekt (Wetter- und Nachrichtenfeeds) über eine MySQL Datenbank in die SAP HANA Datenbank geladen. Das Datenmodell der SAP HANA Komponente besteht daher aus der Zusammenführung der Komponenten *RSS_READER*, *WEATHER_READER* und *ZUG_READER*. Da in SAP HANA die Relationen von Tabellen erst zur Laufzeit gebildet werden, zeigt Abb. 23 eine Übersicht der in SAP HANA verwendeten Tabellen. Die Tabelle *CITIES* steht dabei im Mittelpunkt und enthält die Schlüsselwerte *STATIONID* der *STATIONS* Tabelle, *WOEID* der *WEATHERCITYLIST* und den *CITYNAME* der zur Verknüpfung mit ortsbezogenen Nachrichten dient.

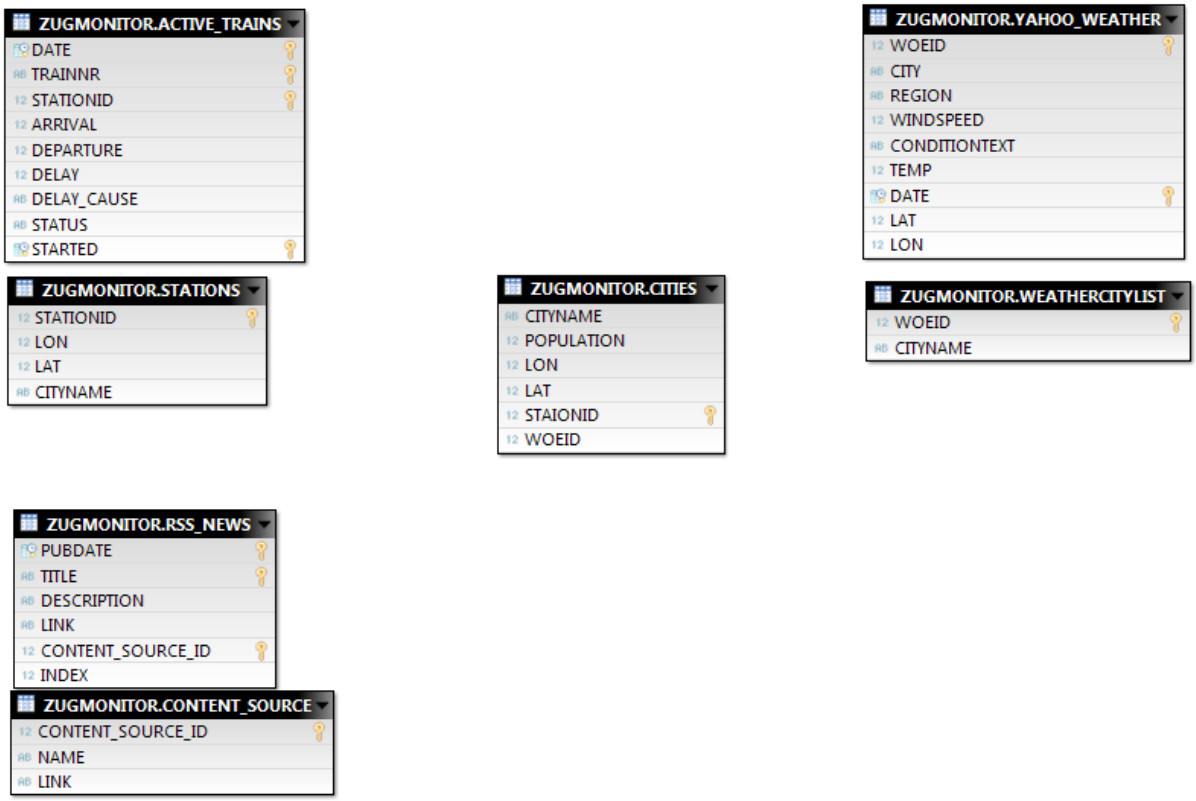


ABB. 23 TABELLEN DER SAP HANA KOMPONENTE

Rss_Reader und Weather_Reader

Das Datenmodell des *RSS_READER*, der Nachrichten und Wetterdaten abruft, besteht im speziellen aus vier Tabellen.

Jeweils eine Konfigurationstabelle, die der Anwendung angibt, welche Daten zu laden sind, und einer Haupttabelle, die die geladenen Daten speichert.

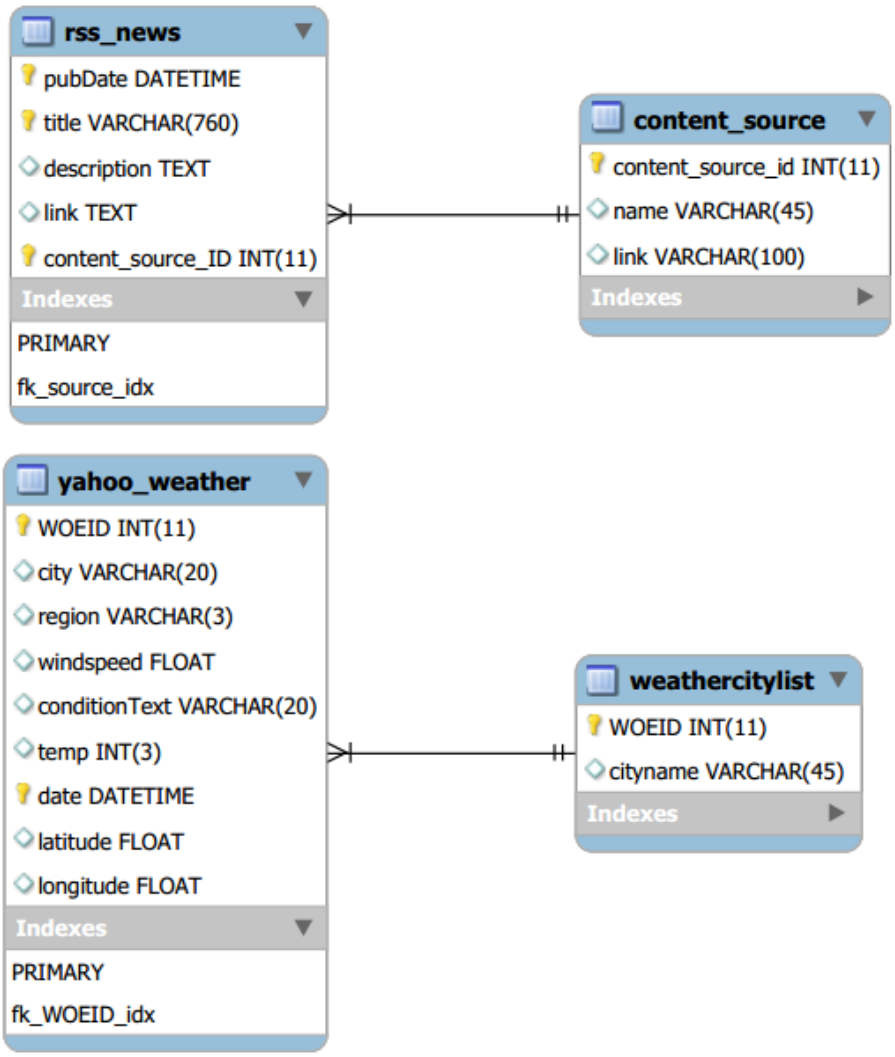


ABB. 24 DATENMODELL DES RSS READERS

Die in Abb. 24 beschriebenen Tabellen werden mittels einer Java Anwendung mit Daten versorgt. Die Java Anwendung selbst wird in regelmäßigen Intervallen auf einem Server ausgeführt und liest dabei RSS Feeds von Yahoo!Weather und Deutschen Zeitungen ausgelesen. Die Selektion erfolgt dabei nach Datum.

Zug_Reader

Die *ZUGMONITOR* Komponente ist prinzipiell an keine spezielle Datenbankschicht gekoppelt, da bereits die Datenquelle *OPENDATACITY* die Daten persistiert. Die Daten liegen in Form von JSON Dateien chronologisch vor. Der in Java implementierte

ZUG_READER wird daher auch benutzergesteuert bei Bedarf gestartet, um die Daten in der Datenbank zu aktualisieren. Das Zieldatenmodell des Readers zeigt Abb. 25.

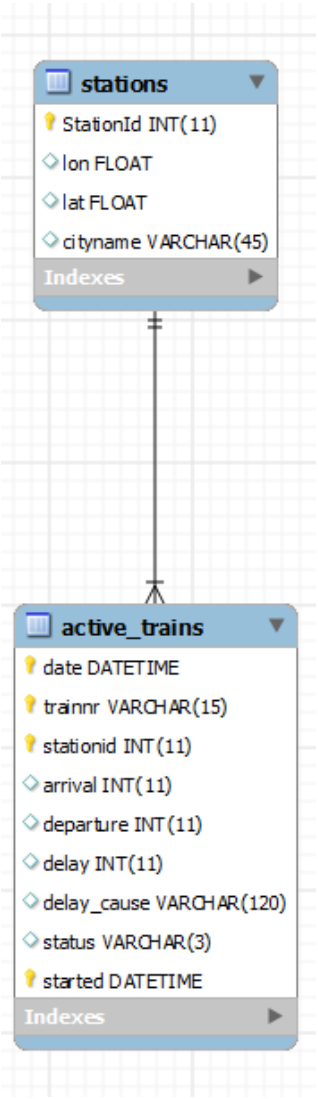


ABB. 25 DATENMODELL DES ZUG_READERS

6.1.3. Ziel der Anwendung

Das vorrangige Ziel der Demoanwendung ist das Aufzeigen ihrer Realisierbarkeit mit der SAP HANA Appliance als Kern Datenbanksystem. Gerade die Entwicklung in der Cloud von Amazon Webservices findet hierbei besondere Betrachtung. Dabei geht es vor allem auch darum, wie sich die Entwicklung in der Cloud von der klassischen offline Entwicklung unterscheidet oder auch ähnelt. Denn wie schon in Kapitel 4 erwähnt, sollte sich die SAP HANA Datenbank für den Anwendungsentwickler im Grunde nur wenig von klassischen Systemen unterscheiden. Mit der Demoanwendung wird zudem von der in SAP HANA zu Verwendung stehenden Möglichkeit Gebrauch gemacht, Business

Logik direkt auf der Datenbankebene auszuführen. Auch dies findet besondere Beachtung.
Die Demoanwendung selbst soll schließlich den Zweck erfüllen, die Fähigkeiten des Systems im Bereich Business Intelligence veranschaulichen. Die Zugverspätungsdaten und Wetterdaten dienen als nummerische Grundlage für Auswertungen von Häufigkeiten und Zusammenhänge. Die für das Data Warehouse üblichen Aggregate, etwa auf Tages oder Monatsebene, sollen dabei nicht gespeichert, sondern stets aus den Rohdaten errechnet werden. Die in reiner Textform vorhandenen Nachrichten erweitern die möglichen Analyseabfragen um unstrukturierte Daten. Fragen, auf die die Anwendung Antworten liefern soll, könnten sein:

- An welchen Bahnhöfen halten die meisten Züge?
- Welche Züge verspäten sich häufig?
- An welchen Bahnhöfen kommt es zu langen Verspätungen?
- Gab es an einem Tag und Ort unregelmäßig viele Verspätungen und kann dies durch einen Zwischenfall (Nachrichten) oder das Wetter erklärt werden?
- Welches sind die häufigsten Verspätungsursachen?

6.2 Bewertung

Die in dieser Arbeit bewerteten Themen sind die SAP HANA Datenbank und dem SAP HANA Studio und deren Einsatz im Business Intelligence. Im speziellen wird dies anhand von Erfahrungen und Erkenntnissen mit dem SAP HANA in der AWS Umgebung und der zuvor beschriebenen Demoanwendung bewertet. Die Umsetzung der Demoanwendung sowie die Bewertung werden in Kapitel 7 erläutert.

6.2.1. Kriterien

Die Kriterien für die Bewertung der SAP HANA Datenbank liegen vor allem im Bereich Entwicklung und Umsetzung von Business Intelligence Projekten mit der SAP HANA Version in der AWS Cloudumgebung. Betriebswirtschaftliche Kriterien wie Lizenzmodelle werden in diesem Kontext nicht bewertet.

Cloudhosting

Hier soll Aufschluss darüber gegeben werden, inwiefern die Entwicklung des SAP HANA Systems in der AWS Cloudumgebung umsetzbar ist. Typische Fragestellung sind die Erreichbarkeit, die Zuverlässigkeit und die Zugriffsgeschwindigkeiten auf die SAP HANA Instanz.

Vergleich mit klassischer Datenbankentwicklung

In der Architektur unterscheidet sich SAP HANA von den meisten relationalen Datenbanksystemen sehr. Mit einer Eclipse-basierten Oberfläche bietet SAP HANA Studio eine Umgebung die vielen Entwicklern vertraut ist. Dieses Kriterium soll bewerten, inwiefern sich Entwickler umstellen oder neu einlernen müssen, um mit SAP HANA zu arbeiten. Wie ist das Verhalten beim Entwickeln von Prozeduren? Wie sind Fehlermeldungen und die entsprechenden Hilfestellungen zur Fehlerbehebung umgesetzt?

Dokumentation

Die Dokumentation ist einer der wichtigsten Bestandteile eines Produktes, um dieses für dritte leicht zugänglich zu machen. Hier soll geklärt werden, wie hilfreich die Dokumentation der SAP HANA Datenbank und deren Entwicklungswerkzeuge sind.

Business Logik im Datenbanksystem

Das SAP HANA Datenbanksystem bietet mit SQLScript die Möglichkeit Business Logik, die normalerweise in anderen Schichten des Anwendungssystems ausgeführt werden, direkt auf Datenbankebene auszuführen. Dieses Kriterium soll bewerten, wie gut dieser Ansatz im speziellen in Verbindung mit der Demoanwendung umgesetzt werden konnte.

Realisierbarkeit

Als übergreifendes Kriterium soll die Realisierbarkeit angeben, inwiefern das in diesem Kapitel entwickelte Anwendungsszenario mit SAP HANA umgesetzt werden konnte.

6.2.2. Schema

Die Bewertung des in dem Anwendungsszenario angewandten SAP HANA Systems erfolgt mithilfe des folgenden Schemas: Jedes Kriterium wird dabei geschlossen für sich betrachtet.

	Schema	**Erläuterung**
Bewertung anhand einer Ordinalskala	0 – 4	0 – Nicht anwendbar 1 – Kriterium nicht erfüllt 2 – Kriterium teilweise erfüllt. 3 – Das Kriterium entspricht der zuvor gestellten Erwartung. 4 – Die zuvor gestellte Erwartung für dieses Kriterium wurde übertroffen
Textuelle Bewertung	Fließtext	Um die Ergebnisse der Bewertungsskala nachvollziehen zu können, werden diese zusätzlich erläutert.

TABELLE 4 BEWERTUNGSSCHEMA

7. Business Intelligence mit SAP HANA in der Amazon Cloud

Wie SAP HANA in Business Intelligence Projekte integriert werden kann, wurde bereits in Abschnitt 4.4 behandelt. Diese Kapitel befasst sich im Speziellen mit der möglichen Umsetzung von Business Intelligence Projekten in der Amazon Cloud. Konkret wird SAP HANA als SAP HANA ONE Instanz bei Amazon Webservices gehostet. Um auf SAP HANA ONE zuzugreifen, wird SAP HANA Studio und Information Composer verwendet.

7.1 Die Entwicklungsumgebung SAP HANA Studio

Wie zuvor erläutert ist die Entwicklungsumgebung für SAP HANA ONE das Eclipse-basierte Tool SAP HANA Studio (vgl. Abschnitt 4.1.2). Im Folgenden wird die erstmalige Konfiguration des HANA Systems und die Datenbankentwicklung für die Cloudbasierte Version HANA ONE erläutert.

7.1.1. Einrichten des SAP HANA Studio

Um mit einer SAP HANA ONE Instanz zu arbeiten, müssen folgende Voraussetzungen erfüllt sein. Es muss sowohl ein Account bei awsmarketplace als auch ein Account für den SAP Marketplace. Sind diese Voraussetzungen erfüllt, kann bei awsmarketplace eine SAP HANA ONE Instanz erstellt werden.

Nach Erstellung der Instanz muss der Zugriff mittels SAP HANA Studio eingerichtet werden. Dazu sind folgende grundsätzliche Schritte nötig.

- Die SAP HANA Instanz muss im Amazon AWS Webportal gestartet werden
- Nach dem Start der Instanz muss sie mit einer *ELASTISCHEN IP* Adresse verknüpft werden. Über diese durch AWS selbst generierte Adresse kann nun auf das Datenbanksystem zugegriffen werden.
- Diese Elastische IP muss (unter Windows) in der *HOSTS* Datei (Pfad unter Windows: C:\Windows\System32\drivers\etc\hosts) mit einem Bezeichner notiert werden. Der Eintrag könnte dann wie folgt aussehen:
    - 54.228.243.61 imdbhdb

  Ohne diese Verknüpfung in der *HOSTS* Datei kann das Datenbanksystem nicht mit SAP HANA Studio gestartet oder gestoppt werden.
- In SAP HANA Studio kann jetzt die Verbindung zu der Instanz eingerichtet werden. Hierzu wird im Navigationsbereich über das Kontextmenü der Eintrag

ADD SYSTEM ausgewählt. Als *HOSTNAME* muss hier dann der Bezeichner aus der *HOSTS* Datei eingetragen werden (hier: imdbhdb).

(vgl. [Schmerder2012])

Nun sollte die Initiale Verbindung zur SAP HANA Datenbank aufgebaut sein und SAP HANA Studio kann zur Entwicklung der Datenbank verwendet werden.

7.1.2. Datenbankentwicklung in der Cloud mit SAP HANA Studio

Zur Erstellung von Datenbankschemata mit SAP HANA stehen dem Entwickler zwei Möglichkeiten zur Verfügung. Zum einen kann der SQL Editor verwendet werden, um SQL Code auszuführen und zum anderen die Funktionalitäten der grafischen Benutzeroberfläche genutzt werden.

SQL Anweisungen

Die SAP HANA Datenbank interpretiert die meisten Standard SQL Anweisungen. Eine Ausnahme ist beispielsweise die Foreign Key Constraint, welche eine Relation zwischen zwei Tabellen herstellt, da die relationale Verknüpfung von Tabellen auf Datenbankebene in SAP HANA nicht möglich ist. Die Ausführung von SQL Skripten aus relationalen Datenbanksystemen, um etwa ein Datenbankschema zu kopieren, ist also nicht ohne weitere Anpassungen möglich.

SQLScript

SQLScript ist eine von SAP entwickelte Erweiterung des SQL Standards. Die Sprache bietet dem Entwickler weitere Sprachelemente, um Anwendungslogik direkt auf Datenbankebene zu verarbeiten. SQLScript bietet beispielsweise die Möglichkeit, Tabellen als Datentypen zu definieren, ohne dass diese auf physische Tabellen in der Datenbank verweisen. Weiterhin können Schleifen und Abfragen, vergleichbar mit anderen Hochsprachen, implementiert werden.

Tabelle 5 zeigt zudem eine Übersicht über SQLScript Funktionen die standardmäßig verwendet werden können. SQLScript wird zudem als wichtiger Bestandteil von *CALCULATION VIEWS* in Abschnitt 7.4.1 behandelt.

SQLScript Command	**Use**
CE_COLUMN_TABLE	Selects data from a columnar table Example: `tab1 = CE_COLUMN_TABLE ("MYTABLE",["FIELD1","FIELD2"]);`
CE_JOIN_VIEW	Selects data from an attribute view. Example: `tab2 = CE_JOIN_VIEW ("ATV_MATERIAL",["MATNR","SPART"]);`

CE_OLAP_VIEW	Selects data from an analytic view.
	Example:
	`tab3 = CE_OLAP_VIEW („ANV_SALESITEM",[ "KUNNR","NETWR"]);`
	`Note that this will provide the SUM of NETWR GROUPed by KUNNR.`

CE_CALC_VIEW	Selects data from a calculation view. Example: `tab4 = CE_CLAC_VIEW ("CAV_SALESITEM",[ "KUNNR","NETWR"]);`
CE_AGGREGATION	Aggregates data, particularly useful for counting. Example: `tab5 = CE_AGGREGATION (:tab3, [COUNT("ID") as "NBR_OF_ORDERS", "KUNNR"]);`
CE_UNION_ALL	Unions two tables. Example: `tab6 = CE_UNION_ALL (:tab4, :tab5);`
CE_JOIN	Performs joins between tables. Example: `tab7 = CE_JOIN (:tab1, :tab2, ["JOINFIELD"],["OUTPUTFIELD1", "OUTPUTFIELD2"]);`
CE_PROJECTION	Apart from restricting the list of columns returned, renames or applies formulas to columns and applies filters. Example: `tab8 = CE_PROJECTION (:tab7, ["KUNNR" as "CUSTOMER", "NETWR" > 100000]);`

TABELLE 5 SQLSCRIPT CE FUNKTIONEN NACH [BERG2012]

Grafische Benutzeroberfläche

Mit SAP HANA Studio können Tabellen und weitere Datenbankelemente auch mittels der grafischen Benutzeroberfläche erstellt werden. Dazu kann das Kontextmenü (vgl. Abb. 26) der Einträge im Navigationsbereich verwendet werden.

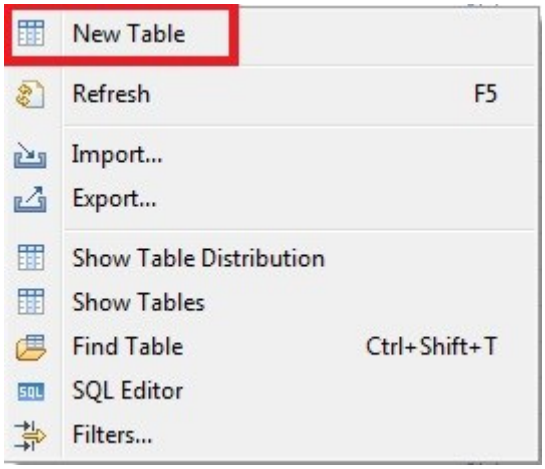


ABB. 26 KONTEXTMENÜ SAP HANA STUDIO ZUR ERSTELLUNG VON TABELLEN

Auch hier ist anzumerken, dass nicht alle Befehle zum Erstellen eines Datenbankschemas abgedeckt sind. Eine neue Schemadefiniton muss beispielsweise über den SQL Editor erstellt werden. Die Tabellen und Views des Schemas können dann mit der grafischen Benutzeroberfläche dem Schema hinzugefügt werden.

7.1.3. Probleme bei der Datenbankentwicklung mit SAP HANA Studio

Während der Umsetzung der Demoanwendung wurden folgende für SAP HANA spezifische Probleme in der Entwicklung festgestellt. Diese Aufführung ist nicht als vollständig anzusehen, sondern stellt lediglich die in dieser Arbeit entstandenen Probleme dar.

Verbindung mit der SAP HANA Datenbank – SAP HANA Studio

Die Instanz der SAP HANA Datenbank war in dieser Arbeit als Cloudumgebung bei AWS gehostet. Die in Abschnitt 7.1.1 beschriebenen Schritte sollten prinzipiell genügen, um eine Verbindung mit der SAP HANA Datenbank aus SAP HANA Studio herzustellen.

Es stellte sich jedoch bei ersten Tests heraus, dass nach einem Neustart der SAP HANA Instanz in der AWS Umgebung Probleme auftreten können. Obwohl die Selbsttests der Instanz die Verfügbarkeit bescheinigen, ist eine Verbindung mittels SAP

HANA Studio nicht immer möglich. Ebenso misslingt in diesem Fall auch eine SSH Verbindung. Wie sich herausstellte ist der Server in diesem Fall nicht mehr über die *ELASTISCHE IP* Adresse erreichbar. Die *ELASTISCHE IP* muss daher nach jedem Neustart der Instanz wieder zugewiesen werden, auch wenn diese laut der AWS Managementconsole bereits schon der Fall sein sollte. Ist eine Verbindung auch jetzt noch nicht möglich, so ist es nötig eine neue *ELASTISCHE IP* Adresse zu generieren und diese mit der SAP HANA Instanz zu verknüpfen. Dabei muss beachtet werden, dass auch der Eintrag in der *HOSTS* Datei geändert werden muss.
Weiterhin kann eine Verbindung per SSH Client erfolgen, um das Datenbanksystem zu verwalten. Das Datenbanksystem kann hiermit beispielsweise neu gestartet werden.

Verbindung mit der SAP HANA Datenbank – Microsoft Excel

Die Anbindung von Microsoft Excel an das SAP HANA System als Reporting Tool ist von SAP vorgesehen und wird daher durch einen entsprechenden Treiber, der in Microsoft Excel verfügbar ist, unterstützt. Erstellte Würfel können in Microsoft Excel jedoch nicht angezeigt werden, wenn diese keine nummerische Kennzahl enthalten. Daher war es in der Demoanwendung nicht möglich Nachrichtenmeldungen zu übertragen, ohne eine künstliche Kennzahl einzufügen.

Unzureichende Rechte

Die Aktivierung von Attribute Views im für diese Arbeit erstellten Schema *ZUGMONITOR* wurde von dem Datenbanksystem anfangs mit der Fehlermeldung *INSUFFICIENT PRIVILEGE: NOT AUTHORIZED* abgewiesen, obwohl der aktuelle Anwender selbst das Schema und die in der Attribute View verwendeten Tabellen erstellt hatte. Zur Lösung des Problems mussten dem Benutzer *_SYS_REPO* die Rechte für das Schema *ZUGMONITOR* übertragen werden.

Verwendung von Zeilen- und Spaltenorientierten Tabellen

Zu Beginn der Arbeit wurde die Möglichkeit der Verwendung von Spalten- und Zeilenorientierten Tabellen getestet. Dabei wurde festgestellt, dass das Verbinden mittels Join-Operationen von Tabellen mit unterschiedlicher Orientierung unter Umständen nicht möglich ist. Daher wurden in dieser Arbeit nur spaltenorientierte Tabellen verwendet.

7.1.4. Datenbankzugriff ohne SAP HANA Studio

Um auf eine SAP HANA Anwendung zuzugreifen können stehen dem Entwickler neben den in Abschnitt 4.4 vorgestellten BI- und ETL Tools weitere Schnittstellen zur Verfügung. Mittels JDBC oder ODBC kann eine Anwendung direkt auf das Datenbanksystem zugreifen und SQL oder MDX Anweisungen ausführen. Die entsprechenden Treiber, die die Anwendung laden müssen, werden zusammen mit der SAP HANA Client Installation im SAP Verzeichnis abgelegt. Die Verbindung via JDBC wird dann beispielsweise mit folgenden Codezeilen hergestellt:

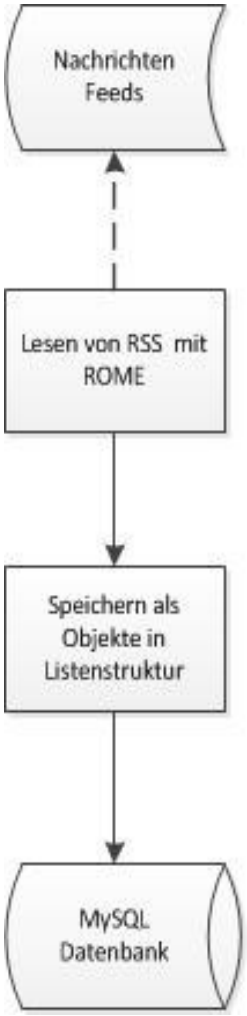
```
Class.forName("com.sap.db.jdbc.Driver");
conn=DriverManager.getConnection("jdbc:sap://imdbhdb:30015?reconnect=true“);
```

LISTING 1 DATENBANKVERBINGUNG MIT JDBC

7.2 ETL Prozesse

Um das SAP HANA Datenbanksystem im Business Intelligence Umfeld entwickeln zu können, war vor der eigentlichen Entwicklung mit SAP HANA Studio die Erstellung von ETL Prozessen nötig. Diese konnten entweder über die in Abschnitt 4.4.1 vorgestellten SAP basierten ETL Tools oder über den Zugriff mit JDBC und Java realisiert werden. Da in dieser Arbeit gezielt die Entwicklung des SAP HANA Datenbanksystems unabhängig von weiteren SAP Produkten betrachtet wird, wurden hier drei Java Anwendungen und eine zusätzliche MySQL Datenbank als Zwischenspeicher implementiert. Das Datenmodell dieser Anwendungen entspricht dem in Abschnitt 6.1.2 vorgestellten. Die drei Java Anwendungen die für das übertragen der Daten von den Quellsystemen in die SAP HANA Datenbank zuständig sind werden nachfolgend vorgestellt.

7.2.1. RSS Reader

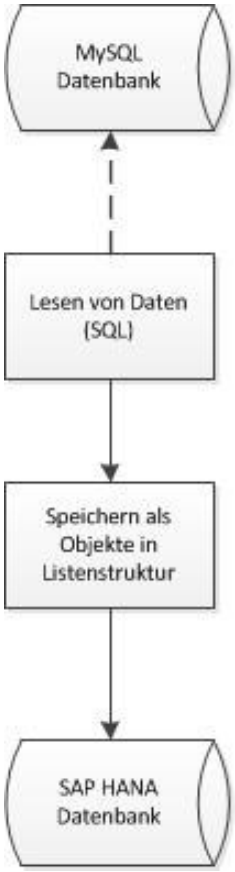


Die Java Anwendung RSS Reader ist für die Persistenz der im RSS Format verfügbaren Nachrichten- und Wetterdaten verantwortlich. Diese Persistenz ist nötig, da die Quelldaten lediglich auf dem aktuellsten Stand auf den Servern der Anbieter verfügbar sind. Bei den Nachrichtenanbietern, wie etwa Spiegel Online, werden die Nachrichten der vergangenen Stunden bereitgestellt. Bei Yahoo!Weather wird sogar nur der aktuellste Wert einer Wetterstation bereitgestellt.

Um also durchgehend die aktuellsten Daten zu sichern, läuft die *RSS_READER* Anwendung auf einem Apache Tomcat Server als Servlet, welches mithilfe der Klassen *java.util.Timer* und *java.util.TimerTask* in 15 Minütigen Intervallen ausgeführt wird. Um die RSS Daten von den entsprechenden Servern zu laden und auszulesen, wird die *ROME* Bibliothek zum Lesen von RSS Feeds verwendet. Die RSS Daten werden in Java Objekte überführt und anschließend über SQL Statements an die MySQL Datenbank gesendet, welche auf demselben Server läuft. Abb. 27 zeigt den Ablauf eines Durchlaufs des RSS Readers.

ABB. 27 ABLAUFDIAGRAMM RSS_READER

7.2.2. RSS_StoreToHANA



Die Java Anwendung *RSS_STORETOHANA* ist als lokale Anwendung konzipiert. Sie dient der Übertragung der Daten aus der MySQL Datenbank, welche die RSS Nachrichten und Wetterdaten enthält, in die SAP HANA Datenbank. Die Anwendung wird durch den Anwender konfiguriert und gestartet. Je nach Konfiguration werden in einer Ausführung der Anwendung Wetterdaten und/oder Zugdaten eines gewählten Zeitraumes übertragen. Für die Übertragung werden die Daten in die gleichen Datenstrukturen geladen wie sie bereits in der *RSS_READER* Anwendung verwendet wurden. Dazu werden Prozeduren der SAP HANA Datenbank verwendet. Abb. 28 zeigt den Ablauf eines Durchlaufs der Anwendung *RSS_STORETOHANA*.

ABB. 28 ABLAUFDIAGRAM RSS STORE TO HANA

7.2.3. Zug_Reader

Da die Zugverbindungsdaten in historisierter Form auf den Servern von *OPENDATACITY* vorliegen, konnte hier auf die Verwendung der MySQL Datenbank als Zwischenspeicher verzichtet werden. Die Aufgabe der *ZUG_READER* Anwendung ist es daher, die Daten aus der Quelle direkt in das SAP HANA System zu laden. Die Zugdaten liegen auf dem Quellserver nach Tagen abgegrenzt im JSON Format vor. Der Aufruf der Daten für einen Tag wird somit über folgenden Link realisiert *HTTP://ZUGMONITOR.SUEDDEUTSCHE.DE/API/TRAINS/<JAHR>-<MONAT>-<TAG>* (Hinweis: Die Daten liegen zwar auf dem Server der Süddeutschen Zeitung. Die API wird jedoch von „Opendatacity“ bereitgestellt und betrieben). Die Quelldaten werden mit der *JACKSON* Bibliothek gelesen und in Java Objekte übertragen. Durch den Aufruf von SAP HANA Prozeduren werden die Zugdaten an die SAP HANA Datenbank gesendet. Abb. 29 zeigt den Ablauf eines Durchlaufs der Anwendung *ZUG_READER*.

ABB. 29 ABLAUFDIAGRAM ZUG READER

7.3 Abbildung der Daten im SAP HANA Datenbanksystem

Die Quelldaten welche mit den in Abschnitt 7.2 erläuterten ETL-Prozessen geladen wurden, werden in SAP HANA in den in Abb. 30 gezeigten Tabellen gehalten.

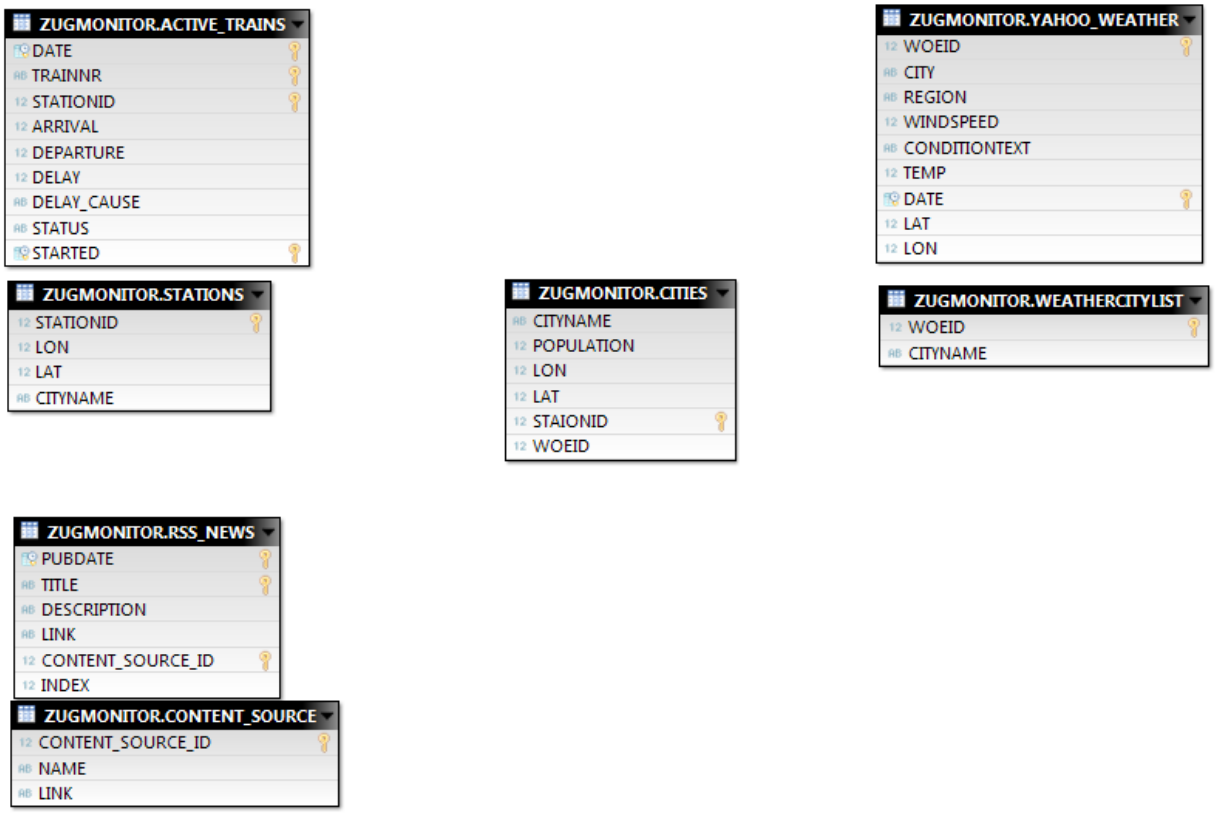


ABB. 30 ÜBERSICHT DER TABELLEN IM SAP HANA DATENBANKSYSTEM

Die Daten werden jeweils in zwei Tabellen gespeichert. Eine Stammdatentabelle und eine Bewegungsdatentabelle. Die Stammdatentabellen enthalten allgemeine Informationen wie den Stadtnamen zu einer entsprechenden ID (*STATIONID* bei Zugdaten und *WOEID* bei Wetterdaten) und der Anbieter zu einer Nachrichten ID (*CONTENT_SOURCE_ID*). Die Bewegungsdatentabellen enthalten bei den Zugdaten (*ACTIVE_TRAINS*) Informationen über die Verspätung (*DELAY*), den Verspätungsgrund (*DELAY_CAUSE*), sowie Ankunft (*ARRIVAL*)- und Abfahrtszeitpunkte (*DEPARTURE*) eines Zuges an einem Bahnhof. Die Wetterdatentabelle (*YAHOO_WEATHER*) enthält vor allem Informationen über die Temperatur (*TEMP*), die Windgeschwindigkeit (*WINDSPEED*) und die textuelle Beschreibung der Wetterbedingungen (*CONDITIONTEXT*) für eine Wetterstation. In der Nachrichtentabelle (*RSS_NEWS*) sind Nachrichtenmeldungen mit Titel (*TITLE*), Beschreibung (*DESCRIPTION*) und Internetlink (*LINK*) abgelegt. Diese Tabelle enthält keine quantitativen Spalten. Da es in SAP HANA keine Fremdschlüssel Attribute gibt, ist auf Tabellenebene keine feste Bindung der Daten erfolgt. Für die

dynamische Verknüpfung der Tabellen durch *ANALYTIC*- und *CALCULATION VIEWS* wird hier zusätzlich die Tabelle *CITIES* verwendet, welche die Schlüsselattribute *STATIONID* und *WOEID* aus Zug- und Wetterdaten enthält und sie über einen gemeinsamen Stadtnamen (*CITYNAME*) verbindet. Die Grundlage des Stadtnamens bildet die Bahnhofsbezeichnung aus den Zugdaten. Die Tabelle *CITIES* kann mit weiteren Attributen wie zum Beispiel hier der Bevölkerungszahl erweitert werden. Die hier verwendeten Einwohnerzahlen wurden aus eine vom Statistischen Bundesamt bereitgestellten Tabelle, die als CSV Datei umgewandelt wurde, eingelesen. Dazu wurde eine kleine Java Anwendung implementiert.

7.3.1. Procedures und SQLScript

Wie im vorangegangenen Abschnitt erwähnt werden Daten zur Speicherung in den Tabellen an SAP HANA Prozeduren (Bezeichnung in SAP HANA Studio *PROCEDURES*) übergeben. Eine Auswahl der *PROCEDURES* wird nachstehend vorgestellt.

Procedure CREATE_WEATHERCITYLIST – Procedure erstellen

Der grundlegende Aufbau einer *PROCEDURE* in SAP HANA soll an dem Beispiel der *PROCEDURE* für das einlesen der Stadtnamen und deren *WOEID*s in die Stammdatentabelle *WEATHERCITYLIST* erläutert werden (Listing 2).

```
CREATE PROCEDURE Zugmonitor.procedure_create_weathercitylist
(in WOEID Integer, in cityname Varchar (45) ) as
BEGIN
INSERT INTO "ZUGMONITOR"."WEATHERCITYLIST" VALUES (WOEID,
      cityname);
 END;
```

LISTING 2 PROCEDURE CREATE_WEATHERCITYLIST

Eine *PROCEDURE* wird in SAP HANA mit dem Schlüsselwort *CREATE* gefolgt von deren Namen erstellt. Die Eingabeparameter werden mit der Schlüsselwort *IN* plus Bezeichnung und Datentyp definiert. Es ist an dieser Stelle auch möglich Ausgabevariablen festzulegen (Schlüsselwort *OUT*). Nach der Definition beginnt die *PROCEDURE* nach dem Schlüsselwort *BEGIN* und wird mit *END* abgeschlossen. Jede Anweisung muss mit einem Semikolon abgeschlossen werden.
In dieser *PROCEDURE* wird lediglich eine SQL Anweisung mit den übergebenen Parametern ausgeführt.

Procedure SACE_ACTIVETRAINS – SQLScript Funktionen

Die *PROCEDURE SACE_ACTIVETRAINS* funktioniert ähnlich wie die zuvor erläuterte *PROCEDURE*. Wie in Listing 3 zu sehen ist, wird hier zusätzlich eine SQLScript Funktion verwendet. *TO_TIMESTAMP* wandelt einen übergebenen String in das gewünschte Datumsformat um.

```
BEGIN insert
INTO "ZUGMONITOR"."ACTIVE_TRAINS" VALUES
      (TO_TIMESTAMP(dateString,'YYYY-MM-DD'),
      trainNr,stId,arrival,departure,delay,delay_cause,status,
      TO_TIMESTAMP(startedString,'YYYY-MM-DD HH:MI:SS'));
END;
```

LISTING 3 PROCEDURE SACE_ACTIVETRAINS (AUSZUG)

Procedure CITIESTOSTATIONS - Kontrollstrukturen

In dieser *PROCEDURE* wird eine weitere Besonderheit von SQLScript gegenüber Standard SQL genutzt (Listing 4). Um die Bahnhofsnamen in Stadtnamen umzuwandeln und mit den Städtebezeichnungen der Wetterdaten in der Tabelle *CITIES* zu speichern wurden zwei For-Schleifen verwendet. Um eine For-Schleife in SQL Script zu verwenden muss zuvor mit dem Schlüsselwort *CURSOR* die Listeneinträge definiert werden. Dieser *CURSOR* muss dann in der For-Schleife verwendet werden. In diesem Beispiel ist auch gut zu erkennen, dass SQLScript und Standard SQL nahezu beliebig zusammen verwendet werden kann. Innerhalb der For-Schleifen werden Standard SQL (*LEFT*) und SQLScript Befehle (*IF*, *THEN*, *ELSE*) verwendet.

```
...
CURSOR cities FOR SELECT * FROM "ZUGMONITOR"."STATIONS";
CURSOR woeids FOR SELECT * FROM "ZUGMONITOR"."WEATHERCITYLIST";

begin FOR cur_row as cities DO foundChar := LOCATE (cur_row.cityname,'(');
      foundSpace := LOCATE (cur_row.cityname,  ' ');

      IF :foundChar <> 0
            THEN trimdcity := LEFT(cur_row.cityname, foundChar-1);
      /*schneidet bei klammern ab*/

            ELSE IF :foundSpace <> 0
                  THEN trimdcity := LEFT(cur_row.cityname, foundSpace-1);
            /*scneidet bei Leerzeichen ab*/
       ...
end;
```

LISTING 4 PROCEDURE CITIESTOSTATIONS (AUSZUG)

Weitere Procedures der Demoanwendung

Die folgende Tabelle zeigt die vollständige Liste der in der Demoanwendung verwendeten *PROCEDURES* sowie deren Funktionalität. Eine *PROCEDURE* wird mit dem Schlüsselwort *CALL* aufgerufen. Dies kann aus dem SQL Editor von SAP HANA, einer anderen *PROCEDURE* oder einer externen Anwendung erfolgen.

Name	**Funktionalität**
PROCEDURE_CREATE_WEATHERCITYLIST	Einlesen der Daten für Stammdatentabelle der Wetterdaten
PROCEDURE_SACE_ACTIVETRAINS	Einlesen der Zugdaten
PROCEDURE_CITIESTOSTATIONS	Generierung der gemeinsamen Stadtnamen und Verknüpfung mit Wetterdaten (WOEID) und Bahnhöfen (STATIONID)
GETCITYNEWS	Liefert alle Nachrichten zu einer übergebenen Stadt
PROCEDURE_SACE_RSS_NEWS	Einlesen der RSS Nachrichten
PROCEDURE_SAVE_STATIONS	Einlesen der Daten für Stammdatentabelle der Zugdaten
PROCEDURE_SAVE_WEATHER	Einlesen der Wetterdaten

TABELLE 6 PROCEDURES DER DEMOANWENDUNG

7.4 Funktionsweise der Demoanwendung

Der primäre Zweck der Demoanwendung ist die Auswertung der Verspätungszeiten der Deutschen Fernverkehrszüge auszuwerten und mit weiteren Informationen aufzubereiten. Hierzu werden wie zuvor beschrieben Wetter und Nachrichtendaten hinzugefügt. Um diese Informationen möglichst auf Datenbankebene aufzubereiten wurden *ANALYTIC*- und *CALCULATION VIEWS* implementiert. Die Ergebnisse aus deren Berechnungen können entweder direkt in SAP HANA Studio in Form einer *PREVIEW* analysiert werden, oder mit externen Business Intelligence Tools als OLAP Würfel, wie hier Microsoft Excel, abgerufen werden. Im Folgenden werden die implementierten *ANALYTIC*- und *CALCULATION VIEWS*, sowie den Zugriff von Microsoft Excel dargestellt.

7.4.1. Analytic- und Calculation Views in SAP HANA

Die nachstehend beschriebenen *ANALYTIC*- und *CALCULATION VIEWS* wurden mit SAP HANA Studio erstellt und greifen auf Daten aus den in Abschnitt 7.2 geladenen Daten zu.

Analytic View – Verspaetungen

Die *ANALYTIC VIEW VERSPAETUNGEN* (Abb. 31) liefert alle Werte aus der Tabelle *ACTIVE_TRAINS*, die im Feld *DELAY* eine Verspätung größer null Minuten aufweist.

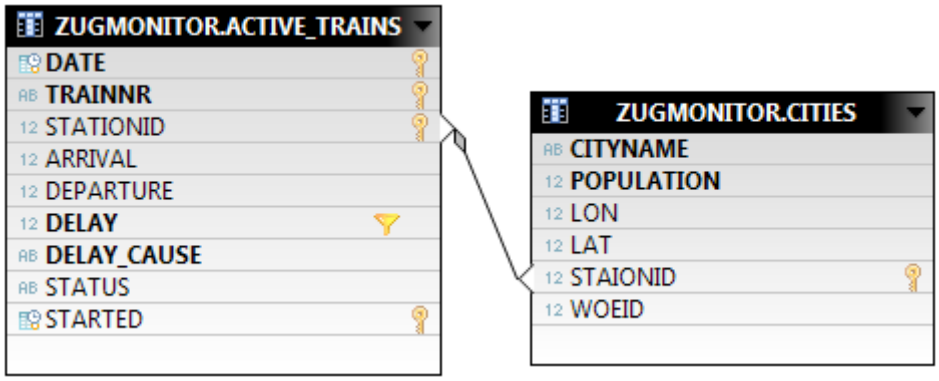
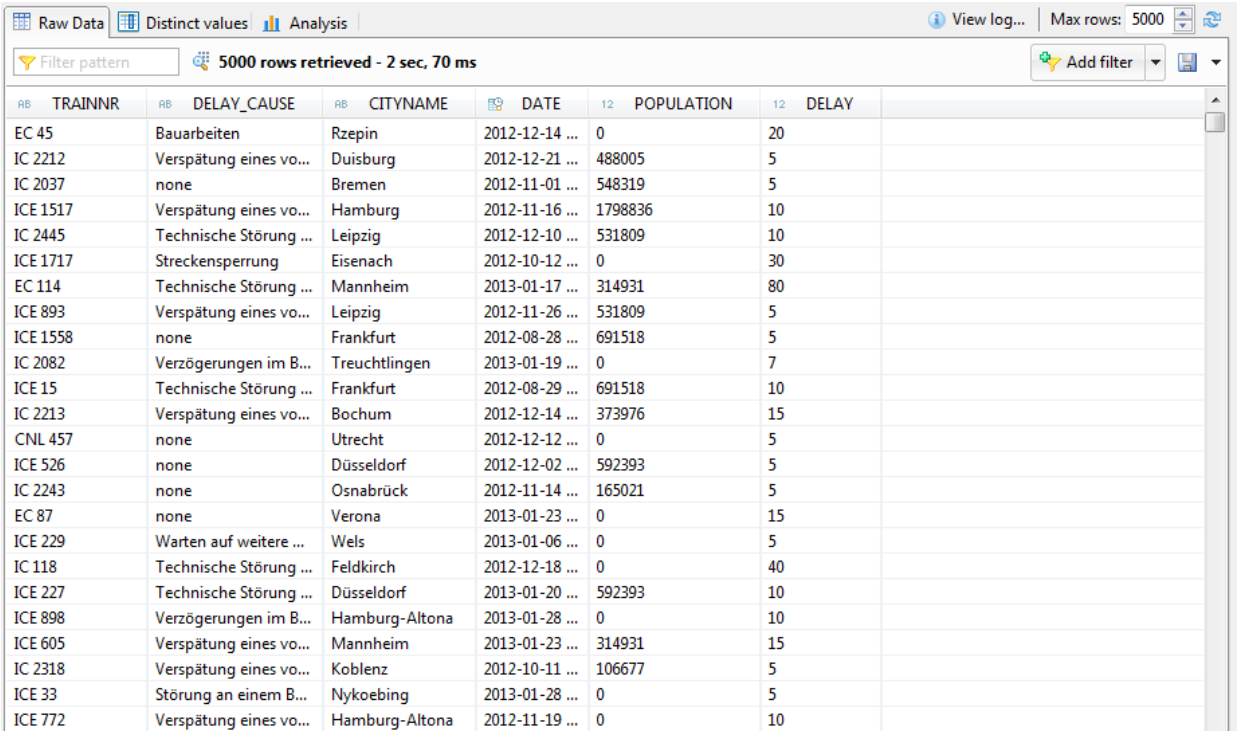


ABB. 31 ANALYTIC VIEW – VERSPÄTUNGEN

Die in der Abbildung Fett Markierten Felder werden als Spalten ausgegeben. Um den Stadtname zu einem Bahnhof zu erhalten wurde die Tabelle *CITIES* referenziert. Zusätzlich kann über die Einwohnerzahl (*POPULATION*) später nach Größe der Städte sortiert werden.

Abb. 32 zeigt die Vorschau dieser View in SAP HANA Studio.



TRAINNR	DELAY_CAUSE	CITYNAME	DATE	POPULATION	DELAY
EC 45	Bauarbeiten	Rzepin	2012-12-14 ...	0	20
IC 2212	Verspätung eines vo...	Duisburg	2012-12-21 ...	488005	5
IC 2037	none	Bremen	2012-11-01 ...	548319	5
ICE 1517	Verspätung eines vo...	Hamburg	2012-11-16 ...	1798836	10
IC 2445	Technische Störung ...	Leipzig	2012-12-10 ...	531809	10
ICE 1717	Streckensperrung	Eisenach	2012-10-12 ...	0	30
EC 114	Technische Störung ...	Mannheim	2013-01-17 ...	314931	80
ICE 893	Verspätung eines vo...	Leipzig	2012-11-26 ...	531809	5
ICE 1558	none	Frankfurt	2012-08-28 ...	691518	5
IC 2082	Verzögerungen im B...	Treuchtlingen	2013-01-19 ...	0	7
ICE 15	Technische Störung ...	Frankfurt	2012-08-29 ...	691518	10
IC 2213	Verspätung eines vo...	Bochum	2012-12-14 ...	373976	15
CNL 457	none	Utrecht	2012-12-12 ...	0	5
ICE 526	none	Düsseldorf	2012-12-02 ...	592393	5
IC 2243	none	Osnabrück	2012-11-14 ...	165021	5
EC 87	none	Verona	2013-01-23 ...	0	15
ICE 229	Warten auf weitere ...	Wels	2013-01-06 ...	0	5
IC 118	Technische Störung ...	Feldkirch	2012-12-18 ...	0	40
ICE 227	Technische Störung ...	Düsseldorf	2013-01-20 ...	592393	10
ICE 898	Verzögerungen im B...	Hamburg-Altona	2013-01-28 ...	0	10
ICE 605	Verspätung eines vo...	Mannheim	2013-01-23 ...	314931	15
IC 2318	Verspätung eines vo...	Koblenz	2012-10-11 ...	106677	5
ICE 33	Störung an einem B...	Nykoebing	2013-01-28 ...	0	5
ICE 772	Verspätung eines vo...	Hamburg-Altona	2012-11-19 ...	0	10

ABB. 32 AUSGABE DER ANALYTIC VIEW – VERSPÄTUNGEN

Die Vorschau gibt die Daten unsortiert als Tabelle aus. Eine Sortierung kann durch Auswählen der Spaltenbezeichnungen erfolgen und es kann über die Schaltfläche *ADD FILTER* nach einem Feld gefiltert werden. Der gesetzte Filter wirkt sich jedoch nicht auf die weiteren in der Vorschau verfügbaren Registerkarten *DISTINCT VALUES* und *ANALYSIS* aus.

Abb. 33 zeigt einen Ausschnitt der Registerkarte *ANALYSIS* mit der eine grafische Auswertung der *ANALYTIC VIEW* möglich ist.

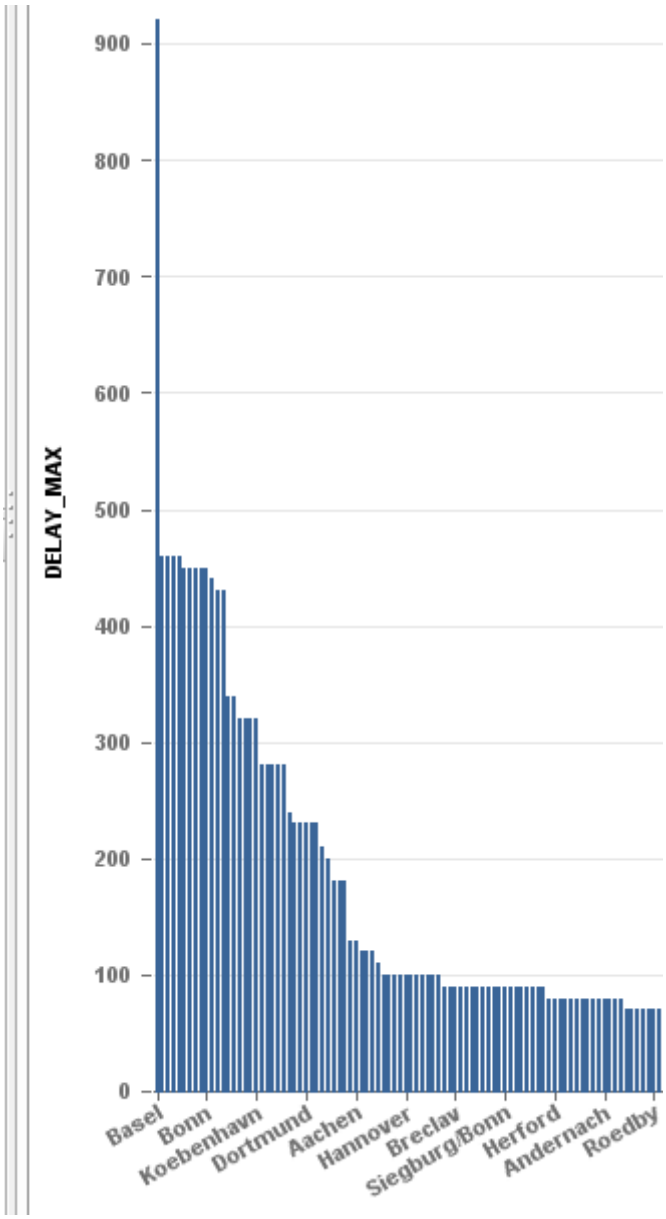


ABB. 33 ANALYTIC VIEW VERSPAETUNGEN - REGISTERKARTE ANALYSIS

Das auf Basis der *ANALYTIC VIEW* dargestellte Balkendiagramm zeigt absteigend sortiert die maximale Verspätung eines Zuges an einem Bahnhof am 20. Januar 2013. Dies könnte ebenso in anderer Form wie etwa einem Kuchendiagramm dargestellt werden. Ein Filter für einen Bereich (etwa für Städte mit Einwohnerzahlen über 100000) ist nicht möglich.

Analytic View – Wetter

Die *ANALYTIC VIEW WETTER* unterscheidet sich zu der View *VERSPAETUNGEN* nur geringfügig. Ähnlich den Verspätungsdaten werden hier die Wetterdaten mit Temperatur, Windgeschwindigkeit und Wetterbedingung ausgegeben. Eine Auswertung mit SAP HANA Studio erfolgt daher analog zu den im vorherigen Abschnitt gezeigten.

Calculation View – Mapcitiesnews

Die in der Demoanwendung verwendeten *CALCULATION VIEWS* wurden mit dem SQLScript Editor und nicht mit dem grafischen Editor erstellt (vgl. Abschnitt 4.1.2 Calculation Views). Eine View ist somit aus dem SQLScript Code (Listing 5) und der Festlegung der Output Variablen definiert.

```
        cur Varchar (30);
        CURSOR c_citycursor FOR
        SELECT "CITYNAME" FROM "ZUGMONITOR"."CITIES";
 BEGIN
        FOR curCity as c_citycursor DO
             cur := curCity.Cityname;
var_temp  =  SELECT  "TITLE","DESCRIPTION","LINK",  "PUBDATE",  :cur  as
CITYNAME
             FROM "ZUGMONITOR"."RSS_NEWS"
             WHERE "TITLE" LIKE CONCAT(CONCAT('%',:cur),'%');
             var_out = CE_UNION_ALL(:var_out, :var_temp);
        END FOR;
END;
```

LISTING 5 SQLSCRIPT DER CALCULATION VIEW MAPCITIESNEWS

In der *CALCULATION VIEW MAPCITIESNEWS* werden zu allen Städten die in der Tabelle *CITIES* hinterlegt sind jeweils mit passenden Nachrichten aus der Tabelle *RSS_NEWS* verknüpft. Dies ist mittels zweier verschachtelter For-Schleifen implementiert, wobei in der äußeren Schleife die Städte und in der inneren die Nachrichten durchlaufen werden.

Die Ergebnisse werden der Variable *VAR_OUT* zugewiesen, welche nach Beendigung des Skriptes an die Output Komponente (vgl. Abb. 34) übergeben wird. Erst in dieser Komponente werden die Attribute, die die View tatsächlich ausgeben soll ausgewählt.

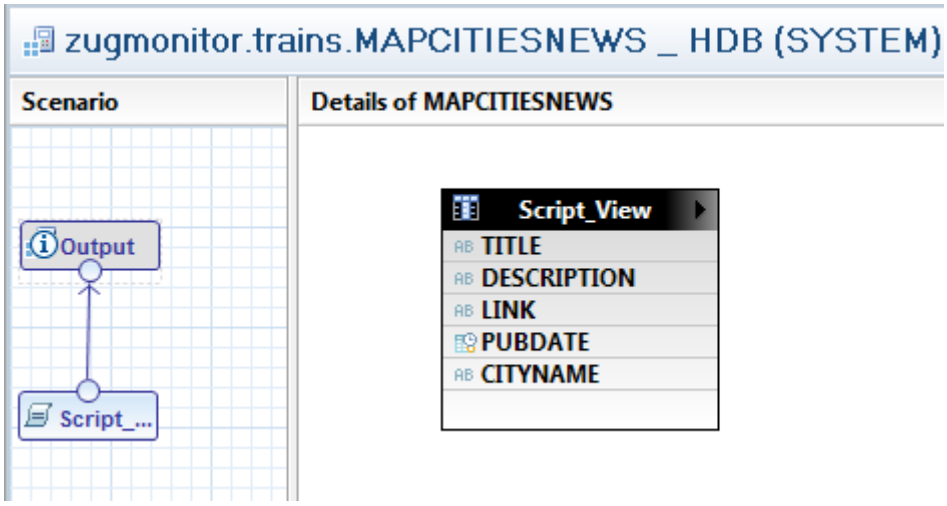


ABB. 34 OUTPUT DER CALCULATION VIEW MAPCITIESNEWS

Die Ausgabe in der SAP HANA Vorschau erfolgt dabei auf gleichem Weg, wie bei *ANALYTIC VIEWS*.

Abb. 35 zeigt beispielsweise die Verteilung von Nachrichtenmeldung anhand der Stadt.

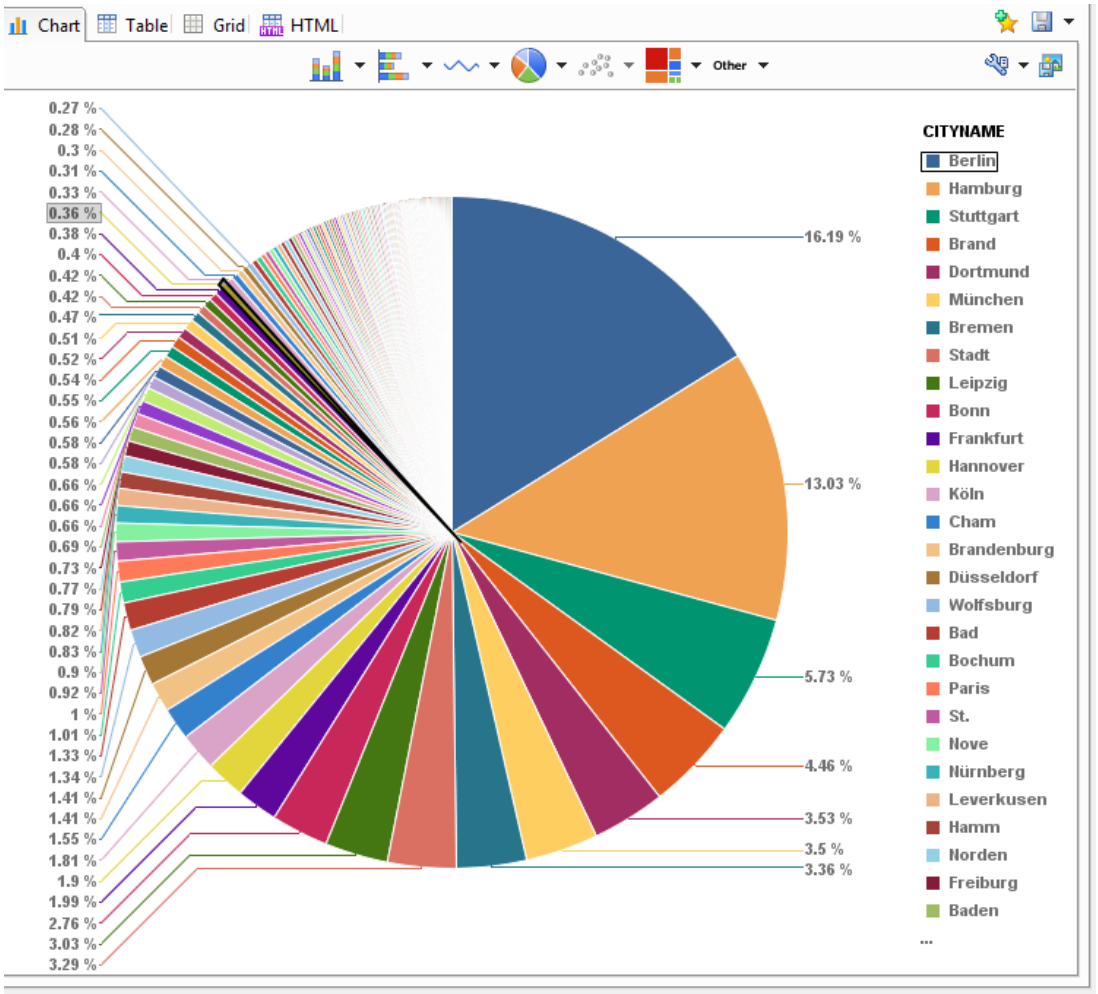


ABB. 35 VERTEILUNG VON NACHRICHTEN NACH STÄDTEN

Calculation View CountTrainsByDate

Diese View dient vor allem zur administration der Zugdaten. Die Ausgabe gibt an wie viele Zugdaten pro Tag in der Tabelle *ACTIVE_TRAINS* enthalten sind. So lässt sich überprüfen ob die Zugdaten vollständig übertragen wurden oder die Quelldaten unvollständig sind. Der SQLScript Teil besteht nur aus einer Standard SQL Anweisung (Listing 6), die eine Übersichtstabelle (Abb.36) erstellt.

```
var_out = SELECT "DATE", COUNT (*) as COUNTER FROM
"ZUGMONITOR"."ACTIVE_TRAINS" GROUP BY "DATE" ORDER BY "DATE";
```

LISTING 6 SQL ANWEISUNG DER CALCULATION VIEW COUNTTRAINSBYDATE

DATE	COUNTER
2013-01-05 00:00:00.0	9199
2013-01-06 00:00:00.0	10036
2013-01-07 00:00:00.0	10240
2013-01-08 00:00:00.0	7471
2013-01-09 00:00:00.0	10136
2013-01-10 00:00:00.0	10190
2013-01-11 00:00:00.0	10961
2013-01-12 00:00:00.0	9049
2013-01-13 00:00:00.0	6860
2013-01-14 00:00:00.0	10337
2013-01-15 00:00:00.0	10171
2013-01-16 00:00:00.0	8543
2013-01-17 00:00:00.0	10195
2013-01-18 00:00:00.0	10979
2013-01-19 00:00:00.0	9087
2013-01-20 00:00:00.0	9955

ABB. 36 AUSGABE DER VIEW COUNTTRAINSBYDATE

7.4.2. Zugriff von Microsoft Excel

Im Folgenden werden der Zugriff und die Verwendung der Analytic- und Calculation Views mit Microsoft Excel dargestellt. In dieser Arbeit wurden die Grundfunktionalitäten von Microsoft Excel verwendet. Die Verwendung des Add-ins PowerPivot, die die Auswertungsergebnisse verbessern würde, konnte aus ungeklärten technischen Gründen (vgl. Abschnitt 7.5) nicht verwendet werden.

Verbindung mit SAP HANA

Hier soll kurz erläutert werden wie die Verbindung zu *ANALYTIC*- und *CALCULATION VIEWS* mit Microsoft Excel hergestellt wird.

Die Verbindung wird als Externe Datenanbindung angelegt und über den Datenverbindungs-Assistenten hergestellt. Dort muss der Datenquelltyp Weitere/erweiterte ausgewählt werden (vgl. Abb. 37).

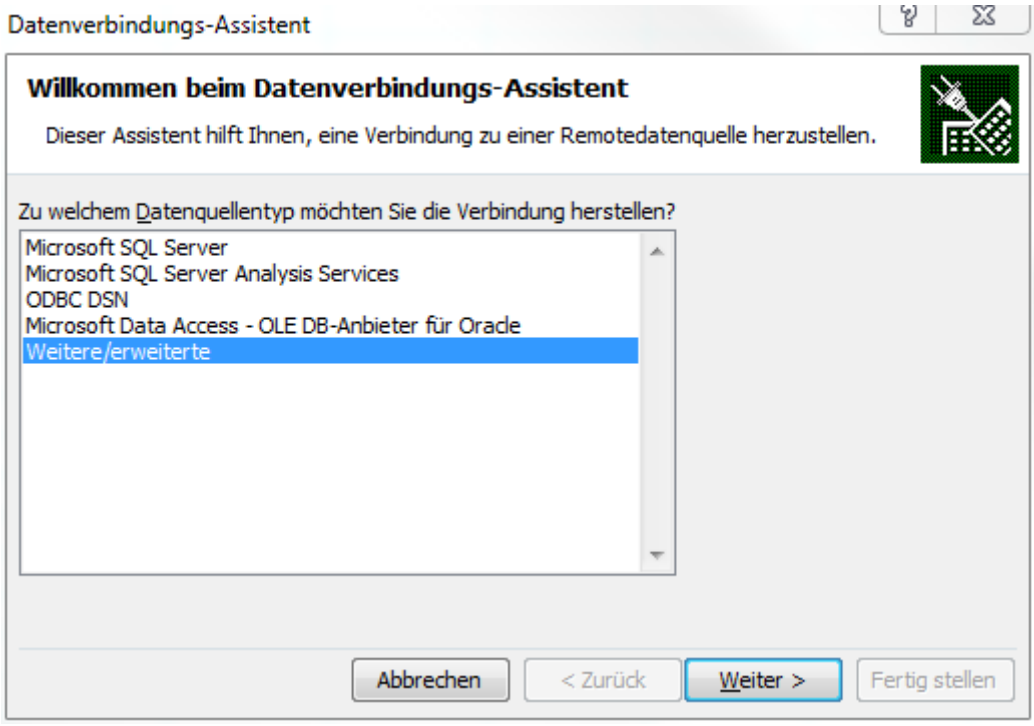


ABB. 37 DATENVERBINUNGS-ASSISTENT

Wurde auf dem System der SAP HANA Client installiert, so wurde mit diesem auch der OLE DB-Provider für SAP HANA installiert und für Microsoft Excel registriert.

Er kann nun als SAP HANA MDX Provider ausgewählt werden (vgl. Abb. 38).

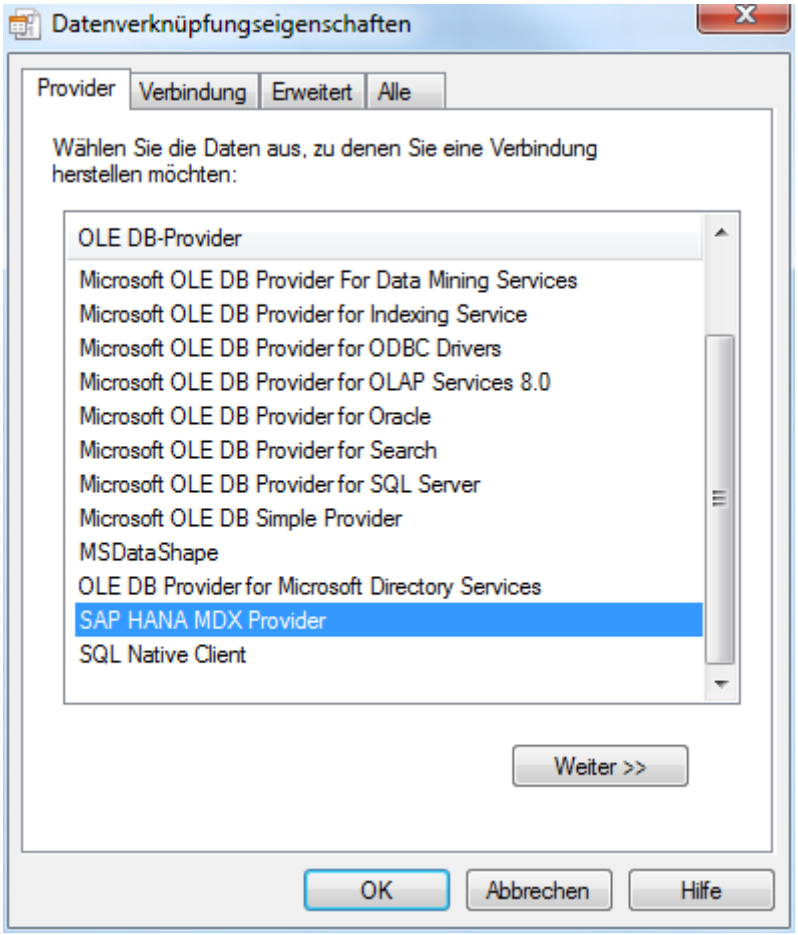


ABB. 38 OLE DB-PROVIDER SAP HANA MDX PROVIDER

Im nächsten Schritt müssen zunächst die Verbindungsdaten für das SAP HANA Datenbanksystem angegeben werden. Diese entsprechen denen, wie sie auch in SAP HANA Studio angelegt wurden.

Ist in der *HOSTS* Datei also die *ELASTISCHE IP* mit dem Wert imdbhdb verknüpft, so kann diese als Hostname verwendet werden(vgl. Abb. 39).

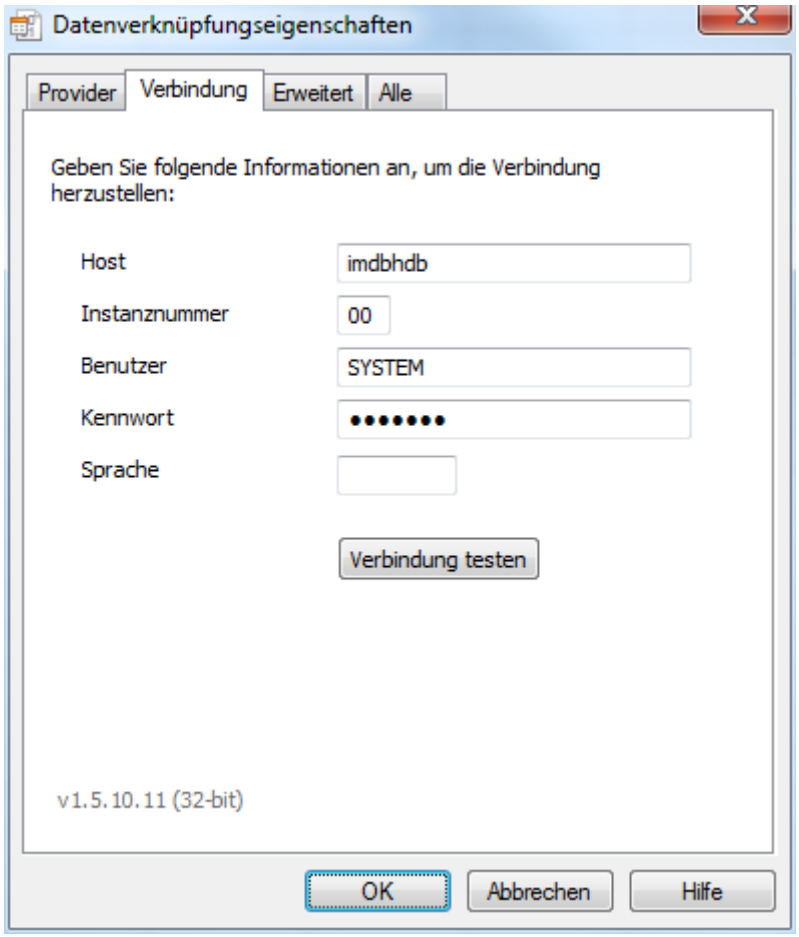


ABB. 39 VERBINDUNGSEIGENSCHAFTEN

Nach herstellen der Verbindung zeigt der Datenverbindungsassistent die Verfügbaren *ANALYTIC*- und *CALCULATION VIEWS* als Cubes an (vgl. Abb. 40). Die Werte, die hinter dem Auswahlfeld für die Datenbank angezeigt werden, entsprechen den in SAP HANA Studio erstellten Paketen. Hier kann beispielsweise auf die *ANALYTIC VIEW VERSPAETUNGEN* und die *CALCULATION VIEW COUNTTRAINSBYDATE* und *MAPCITIESNEWS2* aus dem Paket *ZUGMONITOR.TRAINS* gewählt werden.

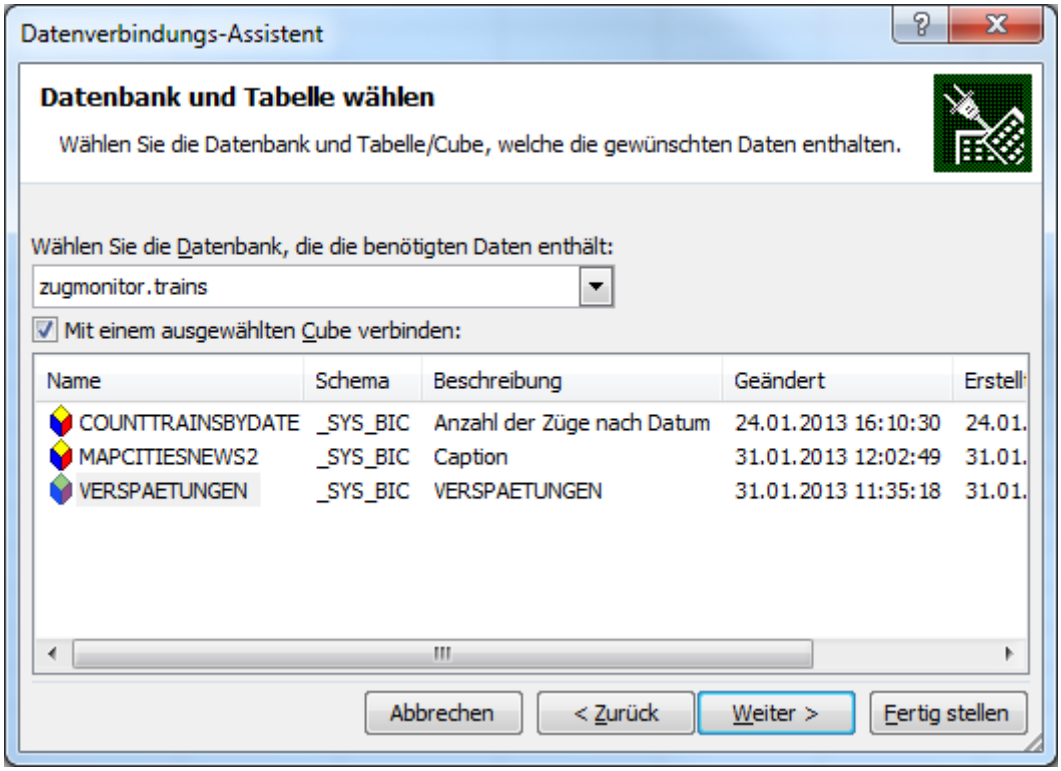


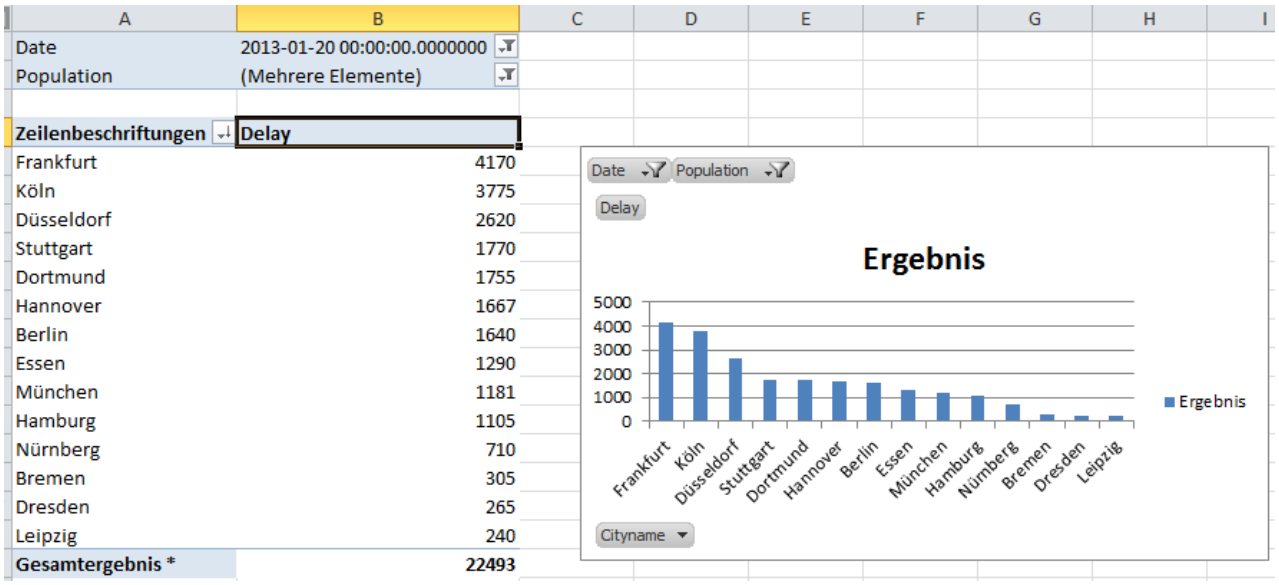
ABB. 40 AUSWAHLDIALOG FÜR ANALYTIC- UND CALCULATION VIEWS

Nach Auswahl der gewünschten View können die Daten in gewohnter Weise mit den PivotTable-Tools von Microsoft Excel verwendet werden.
Die folgenden Abschnitte werden die in der Demoanwendung verwendeten Microsoft Excel Auswertungen vorgestellt. Diese verwenden die in Abschnitt 7.4.1 vorgestellten Views.

Auswertung von Verspätungsdaten nach Städten und Datum

Dieses Szenario verwendet die *ANALYTIC VIEW VERSPAETUNGEN* und vergleicht die Auswertung mit der SAP HANA Studio Vorschau und Microsoft Excel. Die Daten der View lassen sich mit der Standard Pivoting Funktionalitäten modellieren.

Abb. 41 zeigt die Summe aller Verspätungen am 20.01.2013 in Bezug auf die Städte in denen sie auftraten.



	A	B
1	Date	2013-01-20 00:00:00.0000000
2	Population	(Mehrere Elemente)
3		
4	Zeilenbeschriftungen	Delay
5	Frankfurt	4170
6	Köln	3775
7	Düsseldorf	2620
8	Stuttgart	1770
9	Dortmund	1755
10	Hannover	1667
11	Berlin	1640
12	Essen	1290
13	München	1181
14	Hamburg	1105
15	Nürnberg	710
16	Bremen	305
17	Dresden	265
18	Leipzig	240
19	Gesamtergebnis *	22493

ABB. 41 VERSPÄTUNGEN AM 20.01.2013 IN STÄDTEN MIT EINER EINWOHNERZAHL GRÖßER 500000

Im Gegensatz zur Auswertung in der SAP HANA Studio Vorschau werden alle Daten miteinbezogen (in der Vorschau nur 5000 Datensätze) und es besteht die Möglichkeit über die Einwohnerzahlen zu filtern. Nicht möglich ist dagegen die Anwendung einer anderen Aggregatsfunktion als der Summierung.

Auswertung von Ereignissen am Beispiel des Bombenalarms vom 10.12.2012 in Bonn

Dieses Beispiel soll die Verwendungsmöglichkeiten der Demoanwendung aufzeigen. Anhand des Bombenalarms vom 10.12.2012 am Hauptbahnhof in Bonn, der den Zugverkehr erheblich beeinträchtigte, wird die Verwendung verschiedener Views veranschaulicht.

Mit der Verwendung der *CALCULATION VIEW MAPCITIESNEWS* können werden Nachrichten einer gewählten Stadt in einem gewählten Zeitraum ausgegeben. Abb. 42 zeigt einen Auszug der Nachrichtenmeldungen vom 10.12.2012 in Bonn.

Cityname	Bonn
Pubdate	(Mehrere Elemente)
Zeilenbeschriftungen	
Bombenalarm: Bonner Hauptbahnhof evakuiert	
Nach Alarm am Bonner Bahnhof: Zündfähiges Material in Tasche	
Nach Bombenalarm in Bonn: Zugverkehr rollt wieder an	
Gesamtergebnis *	

ABB. 42 NACHRICHTEN VOM 10.12.2012(AUSZUG)

Statt gezielt nach einem bekannten Ereignis zu recherchieren könnte diese View auch verwendet werden um eventuell für den Zugverkehr relevante Ereignisse zu erkennen.
Im nächsten Schritt wurden die Verspätungsursachen analysiert. In den Balkendiagrammen aus Abb. 43 ist deutlich zu erkennen, dass die Verspätungsursache *POLIZEILICHE ERMITTLUNG* für die längsten Verspätungen (in Summe) am 10.12.2012 verantwortlich war.

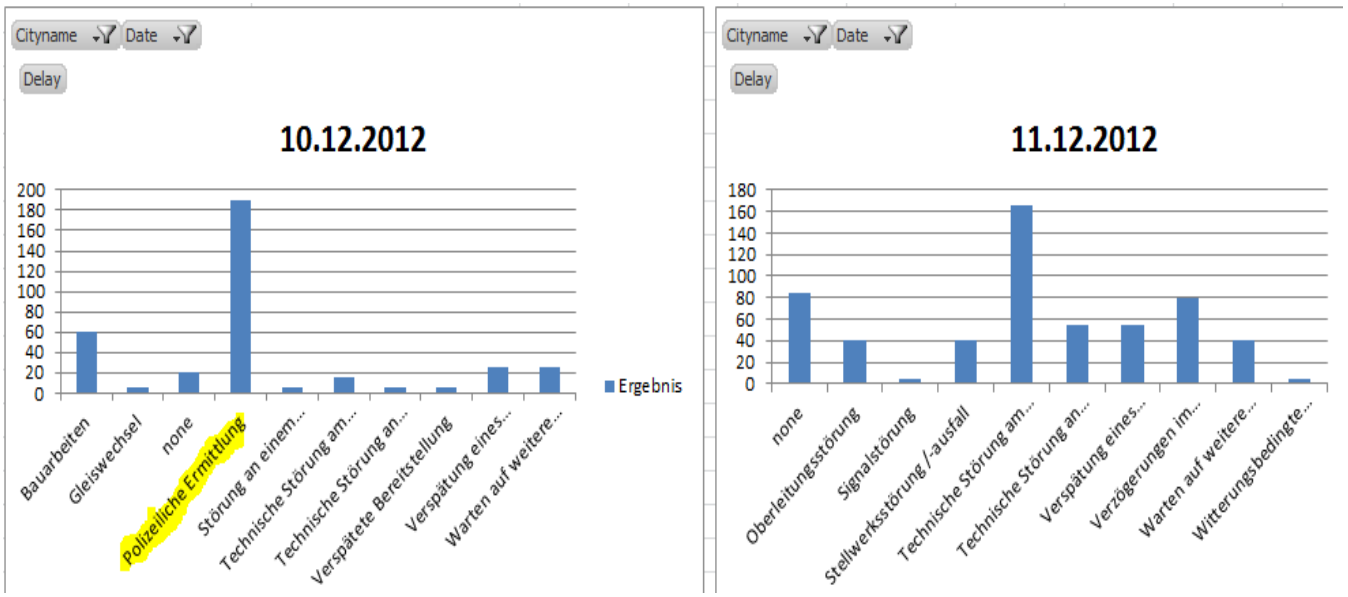


ABB. 43 DIAGRAMM DER VERSPÄTUNGSURCHSACHEN DIE VERZUG AM 10.12.2012 FÜHRTEN

Betrachtet man den Zugverkehr genauer, so ist anzumerken, dass aufgrund dieses Bombenalarms viele Züge erst gar nicht mehr den Bonner Hauptbahnhof anfuhren. Dadurch kam es insgesamt zu weniger Verspätungen als beispielsweise am Folgetag.

7.5 Bewertung der Anwendung in Bezug auf die Kriterien des Anwendungsszenarios

Die Bewertung der Umsetzung der Demoanwendung mit dem SAP HANA Datenbanksystem und SAP HANA Studio werden im Folgenden an den in Abschnitt 6.2.1 festgelegten Kriterien bewertet. Dabei wird das in Abschnitt 6.2.2 beschriebenen Schema angewendet.

7.5.1. Cloudhosting

Die hier Verwendete SAP HANA Instanz läuft vollständig auf den Servern von Amazon Webservices. An dieser Stelle ist anzumerken, dass im Rahmen dieser Arbeit die SAP HANA Instanz nur für die Dauer der aktiven Verwendung aktiv war, weshalb keine Aussage darüber getroffen werden kann wie sich die Instanz im 24 Stunden Betrieb verhält.

Textuelle Bewertung

Nach dem Start der Instanz über das Webportal von Amazon Webservices ist die Instanz verfügbar. Um die Instanz über SAP HANA Studio oder andere Anwendungen zu erreichen, ist die Zuweisung einer Elastischen IP-Adresse nötig. Dieser Schritt ist zwar intuitiv, aber bei jedem Start der Instanz erneut auszuführen. Anschließend ist die Instanz jedoch zuverlässig erreichbar.

Für Demoanwendung hatte dies zur Folge, dass eine weitere Datenbankschicht eingefügt werden musste. Dies lag aber lediglich an dem zeitweisen stoppen der HANA Instanz aus Kostengründen.

Das SAP HANA System steht im Allgemeinen zu jeder Zeit zur Verfügung. Im Zeitraum dieser Arbeit kam es jedoch einmalig zu einem Datenverlust auf einem Datenträger auf dem Informationen dieser Instanz gespeichert waren. In der Folge war die Instanz nicht mehr zu Starten und es musste eine neue SAP HANA Instanz erstellt werden.

Die Zugriffsgeschwindigkeit auf die SAP HANA Instanz wurde nicht explizit getestet. Sie entsprach jedoch stets den Erwartungen.

Bewertung anhand der Bewertungsskala

Auf der im Schema vorgegeben Skala von 0-4 wird dieser Punkt mit 2 bewertet, aufgrund der leichten Komplikationen in der Zuverlässigkeit und Erreichbarkeit.

7.5.2. Vergleich der Entwicklungsumgebung mit klassischer Datenbankentwicklung

Der Vergleich gegenüber relationaler Datenbanksysteme ist vor allem daher interessant, da SAP HANA Studio Daten in spaltenorientieren Strukturen speichert.

Textuelle Bewertung

Bedenkt man den starken architektonischen Unterschied zwischen spalten- und zeilenorientierten Datenbanksystemen, so sind die Unterschiede in der Datenbankentwicklung gering. Das Erstellen von Tabellen mit der grafischen Benutzeroberfläche in SAP HANA Studio ähnelt bekannten Mustern. Es spielt an dieser Stelle dann auch keine Rolle ob Tabellen als spalten- oder zeilenorientier angelegt werden. Die Verwendung von beiden Tabellentypen führt erst bei der Verwendung von Views zu Problemen, weshalb in der Demoanwendung auch nur spaltenorientierte Tabellen verwendet wurden.

Die SAP HANA Datenbank interpretiert sowohl SQLScript als auch Standard SQL, wobei nicht alle Standard SQL Befehle angewendet werden können. Gerade die Tatsache, dass SAP HANA keine Fremdschlüsselbeziehungen beim Erstellen von Tabellen zulässt verkompliziert die Anwendung von bestehenden SQL Skripten zur Erstellung des Datenbankschemas.

Die ETL Prozesse der Demoanwendung können zudem zuverlässig auf *PROCEDURES* zugreifen und direkt SQL Anweisungen ausführen.

Erwartungsgemäß sind auch die Fehlermeldungen die sowohl bei Fehler in SQLScript als auch bei Fehler in SQL Befehlen abgesetzt werden.

Bei der Entwicklung von *CALCULATION VIEWS* mit dem SQLScript Editor wäre jedoch ein Debugging Modus hilfreich, welcher in der in dieser Arbeit verwendeten Version noch nicht zur Verfügung stand.

Generell ist das Erstellen von *CALCULATION VIEWS* mühsam. Es kann vorkommen, dass eine bereits aktivierte View nach korrekten Änderungen nicht mehr aktiviert werden kann mit der Meldung, dass diese bereits existiert. Um dieses Problem zu beheben muss eine neue *CALCULATION VIEW*, auf Basis der aktuellen, angelegt werden. Zudem muss eine neue Bezeichnung verwendet werden, da die aktuelle View nicht vollständig aus dem System gelöscht werden kann.

Für die Demoanwendung bedeuten die Probleme der *CALCULATION VIEW*, die vor allem für die Analyse der Daten verwendet wurden, einen erheblichen Mehraufwand.

Bewertung anhand der Bewertungsskala

Auf der im Schema vorgegeben Skala von 0-4 wird dieser Punkt mit 3 bewertet, da die Erwartungen an die Datenbankentwicklung erfüllt wurden und lediglich die Bereiche *CALCULATION VIEW* und Debugging negativ zu bewerten sind.

7.5.3. Dokumentation

Das Kriterium der Dokumentation umfasst sowohl Online als auch Printmedien die zur Erstellung der Demoanwendung verwendet wurden.

Textuelle Bewertung

Für die Erstellung dieser Arbeit und der Demoanwendung war das Buch „SAP HANA – An Introduction“ [Berg2012], das PDF Dokument „SAP In-Memory Database – SQLScript Guide“ und die Onlinedokumentation „SAP HANA Reference“ hilfreich.
Während das Buch Aufschluss über Grundlagen und Zusammenhänge mit weiteren Komponenten gibt, enthalten die weiteren genannte Dokumente hauptsächlich Informationen zu Datentypen und SQLScript. Beide Dokumente erheben jedoch scheinbar keinen Anspruch auf Vollständigkeit. In der SAP HANA Reference fehlt beispielsweise die Erläuterung der SQL Script Funktionen. Eine explizite Dokumentation über die Entwicklungsumgebung SAP HANA Studio ist dagegen nur ansatzweise im Buch enthalten. Weitere Informationen konnten auch in den SAP Foren recherchiert werden.

Bewertung anhand der Bewertungsskala

Auf der im Schema vorgegeben Skala von 0-4 wird dieser Punkt mit 2 bewertet, da die Dokumentation an sich zwar hilfreich aber nicht vollständig ist.

7.5.4. Business Logik im Datenbanksystem

Da SAP HANA darauf ausgelegt große Teile der Business Logik auf Datenbankebene auszuführen wird dieser Punkt nochmals behandelt, auch wenn *CALCULATION VIEWS* und SQL Skript bereits im Umfeld der Entwicklungsumgebung bewertet wurde.

Textuelle Bewertung

Vor allem mit *CALCULATION VIEWS*, welche mit SQLScript erstellt werden bietet SAP HANA viele Möglichkeiten um komplexe Berechnungen auf Datenbankebene auszuführen. Zum Sprachumfang von SQL Skript gehören unter anderem Kontrollstrukturen und vordefinierte Funktionen die den Code übersichtlicher machen können. Der bereits erwähnte Debugging Modus sollte, sofern in der entsprechenden Version vorhanden, die Entwicklung zudem deutlich vereinfachen. Dass die In-Memory Technologie nicht per Definition schnell sein muss, zeigt dagegen die in der Demoanwendung verwendete View *MAPCITIESNEWS*. Die Laufzeit liegt derzeit bei über zehn Sekunden, was jedoch durch die hohe Anzahl an SQL abfragen liegen sollte.

Bewertung anhand der Bewertungsskala

Auf der im Schema vorgegeben Skala von 0-4 wird dieser Punkt mit 3 bewertet, da abgesehen von den Problemen welche bei der Entwicklungsumgebung zu Abzug führten, *CALCULATION VIEWS* und SQLScript die Erwartung erfüllen konnte.

7.5.5. Realisierbarkeit der Demoanwendung

In diesem Kriterium wird SAP HANA in Bezug auf die Realisierung mit der Demoanwendung bewertet.

Textuelle Bewertung

In der Demoanwendung hat das SAP HANA Datenbanksystem den Charakter eines Data Warehouses. Die Daten wurden entweder direkt aus den Quellsystemen oder über eine weitere Persistenz Schicht in das System geladen. Daher ist sind die ETL Prozesse eine wichtige Komponente die für SAP HANA konfiguriert werden müssen. Da neben dem SAP HANA Datenbanksystem keine weitere SAP Produkte verwendet werden, ist das Laden der Daten nur über JDBC/ODBC und SQL möglich. In dieser Arbeit wurden dafür Java Anwendungen (vgl. Abschnitt 7.2) entwickelt, welche Teilweise auf *PROCEDURES* des SAP HANA Datenbanksystems zugreifen. Durch die Verwendung von SAP Produkte wie etwa Data Services wäre die Entwicklung vermutlich leichter gewesen.

Eine weitere kritische Komponente des Systems ist das Business Intelligence Tool zur Analyse der Daten. Auch hier wird kein SAP Produkt wie etwa Business Objects

eingesetzt, sondern Microsoft Excel. Dieses Produkt ist zwar ebenfalls von SAP für die Verwendung mit HANA zertifiziert, eignete sich aber in der Verwendung in dieser Arbeit nur bedingt. Der Einsatz des Add-ins PowerPivot war beispielsweise nicht möglich. Microsoft Excel beendete sich beim Verbindungsaufbau zwischen PowerPivot und SAP HANA ohne Fehlermeldung.
Die Entwicklung der Schemata, *PROCEDURES* und Views innerhalb von HANA funktionierte dagegen weitgehend problemlos (siehe auch Abschnitt 7.5.2).

Bewertung anhand der Bewertungsskala

Auf der im Schema vorgegeben Skala von 0-4 wird dieser Punkt mit 2 bewertet, da die Realisierung der Demoanwendung ohne Verwendung weiterer SAP Produkte schwerer umzusetzen war, als sie es möglicherweise mit anderen Datenbanksystemen gewesen wäre.

8. Fazit

Diese Arbeit beschäftigte sich zunächst mit der Analyse der derzeitigen Gegebenheiten und aktuellen Erkenntnissen im Bereich Business Intelligence. So erweckte gerade die Trennung der Datenbanksysteme für OLTP und OLAP besonderes Interesse (vgl. Abschnitt 2.5.3). Ausgehend von den technischen Gegebenheiten in der Vergangenheit, war durch den Schritt der Teilung der Systeme zum damaligen Zeitpunkt eine Leistungssteigerung möglich, welche aus heutiger Sicht jedoch neu evaluiert werden muss. Dagegen muss jedoch auch festgehalten werden, dass heutige Data-Warehouse Systeme nicht nur die operativen Systeme entlasten, sondern auch die gemeinsame Basis für Daten aus verschiedenen internen und externen Quellsystemen bilden. Bei einer Zusammenführung von OLTP und OLAP in einer Datenschicht, muss das verwendete Datenbanksystem für die Quellsysteme entsprechend angepasst werden und Data-Warehouse Strukturen unter Umständen auf einer logischen Schicht abbilden. Das In-Memory Datenbanksystem SAP HANA versucht mit dessen Architektur zumindest die Voraussetzungen für die Zusammenführung von OLTP und OLAP zu bieten. Das Datenbanksystem bildet alle Daten in spaltenorientierten Strukturen ab und verwendet darüber hinaus den Hauptspeicher als Kernspeichermedium. Festplatten haben lediglich eine backup-Funktion. Aufgrund dieser Eigenschaften allein ist SAP HANA als OLAP System geeignet, da es enorme Geschwindigkeitsvorteile bei analytischen Abfragen bietet. Wie Abschnitt 4.1.1 zeigt bietet das Datenbanksystem auch Eigenschaften, die für OLTP geeignet sind. Sämtliche Insert oder Update Anweisungen werden intern in schreiboptimierten zeilenorientierte Tabellen verarbeitet und erst später in die leseoptimierte Haupttabellen übertragen (vgl. Abb. 12, Abschnitt 4.1.1). Die Verwendung von *ANALYTIC*- und *CALCULATION VIEWS* zur Erstellung von OLAP-Würfeln, sowie SQLScript zur Implementierung von Business Logik bieten Ansatzmöglichkeiten um OLTP und OLAP allein auf dem Quellsystem auf Basis von SAP HANA umzusetzen.

Im praktischen Teil dieser Arbeit wurde eine Demoanwendung entwickelt, welche Verspätungsdaten der Deutschen Fernverkehrszüge in einer Business Intelligence typischen Weise mit Wetterdaten und textuellen Nachrichtendaten verknüpft. Als Hauptdatenbanksystem dient hierzu ein SAP HANA System in einer AWS Cloudumgebung. Zunächst stand dabei die Entwicklung der ETL Prozesse im Vordergrund, welche aufgrund des Verzichtes auf den Einsatz von weiteren SAP Produkten, mit Java, SQL und JDBC umgesetzt wurde. Daher wurde auch das

Datenmodell relativ flach mit wenigen Tabellen konzipiert und Daten wurden nur geringfügig transformiert. In der Entwicklung von *ANALYTIC*- und *CALCULATION VIEWS* führte diese Entscheidung später zu unerwarteten Schwierigkeiten, was andererseits aber auch teilweise den Hindernissen der Entwicklung von *CALCULATION VIEWS* geschuldet ist (vgl. Abschnitt 7.5.3). Der zur Verfügung stehende Funktionsumfang von Microsoft Excel in Verbindung mit *ANALYTIC*- und *CALCULATION VIEWS* war ebenfalls nur bedingt verwendbar.

Das SAP HANA Datenbanksystem ist abschließend betrachtet allein schon aus Sicht der technischen Voraussetzungen den Einsatz im Business Intelligence Wert, wie auch schon einige praktische Einsätze zeigen. In der Cloud als SAP HANA ONE ist es zudem intuitiv als Datenbanksystem Einsetzbar und unterscheidet sich in der Datenbankentwicklung nur wenig gegenüber der Entwicklung auf einem eigenen Server. Diese Arbeit hat jedoch gezeigt, dass der Mehrwert von SAP HANA für sich alleine derzeit eher gering ist. In Verbindung mit weiteren SAP Produkten wie dem Sybase Replication Server um Daten in Echtzeit zu Laden oder SAP Business Objects als Business Intelligence Tool ist SAP HANA effektiv einsetzbar.

Literaturverzeichnis

[BaGü2009] Bauer Andreas, Günzel Holger (2009). Data-Warehouse-Systeme. dpunkt.verlag

[Berg2012] Berg Bjarne, Silvia, Penny (2012). SAP HANA. SAP Press.

[SBHD1998] Sapia Carsten, Blaschka Markus, Höfling Gabriele, Dinter Barbara (1998). Extending the E/R Model for the Multidimensional Paradigm. Verfügbar als PDF: http://www.fing.edu.uy/inco/grupos/csi/esp/Cursos/cursos_act/2003/DAP_SistDW/Material/sap99a.pdf

[Färber2011] Färber Franz, Cha Sang Kyun, Primsch Jürgen, Bornhövd Christof, Sigg Stefan, Lehner Wolfgang (2011). SAP HANA Database - Data Management for Modern Business Applications. Ausgabe 4/40. SIGMOD Record

[Plattner2012] Plattner Hasso (2012). A Common Approach for OLTP and OLAP Using an In-Memory Column Database. Hasso Plattner Institute for IT Systems Engineering.

[Word2012] Word Jeffrey (2012). SAP HANA Essentials. Epistemy Press LLC

[KrJo2012] Krüger Jens (2012). Hauptspeicherdatenbanken für Unternehmensanwendungen. 11. Berlin-Brandenburger SAP-Forum

[Sigg2012] Sigg Stefan(2012). SAP HANA – Overview and Usage Potential. 11. Berlin-Brandenburger SAP-Forum

[Schmerder2012] Schmerder Juergen (2012). Get your own SAP HANA ONE. Verfügbar als HTML: http://scn.sap.com/docs/DOC-28294

[KBM2010] Kemper Hans-Georg, Baars Henning, Mehanna Walid (2010). Business Intelligence - Grundlagen und praktische Anwendungen. Vieweg+Teubner

[Opendatacity2012] Opendatacity. Opendatacity – Zugmonitor API (2012). Verfügbar als HTML: http://www.Opendatacity2012.de/

[PlZe2011] Plattner Hasso, Zeier Alexander (2011). In-Memory Data Management. Springer

[SFLCPB2011] Sikka Vishal, Färber Franz, Lehner Wolfgang, Kyun Cha Sang, Peh Thomas, Bornhövd Christof (2011). Efficient Transaction Processing in SAP HANA Database - The End of a Column Store Myth. Ausgabe 4/40. SIGMOD Record

Abbildungsverzeichnis

Abb. 1 Business Intelligence – Architektur [KBM2010] 11
Abb. 2 Architektur bei Verwendung abhänger Data Marts. nach [BaGü2009] 17
Abb. 3 Architektur bei Verwendung unabhängiger Data Marts. nach [BaGü2009] 18
Abb. 4 OLAP Würfel [BaGü2009] 21
Abb. 5 Snowflake-schema 23
Abb. 6 Star-Schema 24
Abb. 7 Workloads nach Anwendungsfall im Vergleich zum TPC-C Benchmark [KrJo2012] 27
Abb. 8 Anordnung von Daten im Spaltenorientierten Datenbanksystemen 30
Abb. 9 Anordnung der Daten in einem Spaltenorientierten DBMS 31
Abb. 10 Vergleich des Zugriffs bei relationalen- und spaltenorientierten DBMS[PlZe2011] 32
Abb. 11 Lebenszyklusmanagement von Datenbankeinträgen[SFLCPB2011] 37
Abb. 12 Navigator 38
Abb. 13 Erstellen einer Tabelle mit dem grafischen Editor 39
Abb. 14 Join über zwei Tabellen mit Visual SQL 39
Abb. 15 Content Ordner im Navigations Fenster 40
Abb. 16 Attribute view - Referentieller Join 41
Abb. 17 Analytic View 42
Abb. 18 Grafischer editor für calculation view 43
Abb. 19 Calculate View 43
Abb. 20 Wiederherstellung nach Stromausfall. nach [Berg2012] 47
Abb. 21 SAP HANA IM Business Intelligence Umfeld [Färber2011] 48
Abb. 22 Komponentenarchitektur der Demoanwendung 60
Abb. 23 Tabellen der SAP HANA Komponente 61
Abb. 24 Datenmodell des RSS REaders 62
Abb. 25 Datenmodell des Zug_Readers 63
Abb. 26 Kontextmenü SAP HANA Studio zur erstellung von Tabellen 70
Abb. 27 Ablaufdiagramm RSS_Reader 73
Abb. 28 Ablaufdiagram RSS Store to HANA 74
Abb. 29 Ablaufdiagram Zug reader 75
Abb. 30 Übersicht der Tabellen im SAP HANA Datenbanksystem 76
Abb. 31 Analytic view – Verspätungen 80
Abb. 32 Ausgabe der Analytic View – Verspätungen 81
Abb. 33 Analytic View Verspaetungen - Registerkarte Analysis 82
Abb. 34 Output der Calculation View Mapcitiesnews 84
Abb. 35 Verteilung von Nachrichten nach Städten 85
Abb. 36 Ausgabe der VIEW Counttrainsbydate 86
Abb. 37 Datenverbinungs-Assistent 87
Abb. 38 OLE DB-Provider SAP HANA MDX Provider 88
Abb. 39 Verbindungseigenschaften 89
Abb. 40 Auswahldialog für ANalytic- und Calculation views 90
Abb. 41 Verspätungen am 20.01.2013 in Städten mit einer einwohnerzahl Größer 500000 91
Abb. 42 Nachrichten vom 10.12.2012(Auszug) 92
Abb. 43 Diagramm der Verspätungsurchsachen die verzug am 10.12.2012 führten 92

Tabellenverzeichnis

Tabelle 1 Vergleich OLTP und OLAP nach [PlZe2011] 26
Tabelle 2 Datensätze der Beispieltabelle 31
Tabelle 3 Hardware Konfigurationsoptionen für SAP HANA nach[Berg2012] 44
Tabelle 4 Bewertungsschema 66
Tabelle 5 Procedures der Demoanwendung 79

Listingverzeichnis

Listing 1 Datenbankverbingung mit JDBC 72
Listing 2 Procedure_CREATE_WEATHERCITYLIST 77
Listing 3 PROCEDURE_SACE_ACTIVETRAINS (Auszug) 78
Listing 4 PROCEDURE_CITIESTOSTATIONS (Auszug) 78
Listing 5 SQLScript der Calculation View MapCitiesNews 83
Listing 6 SQL Anweisung der Calculation VIEW Counttrainsbydate 86

Abkürzungsverzeichnis

BI = Business Intelligence
DWH = Data Warehouse
CRM System = Customer Relationship Management System
ETL = Extract, Transform, Load
OLAP = Online Analytical Processing
SQL = Structured Query Language
MDX = Multidimensional Expressions
OLTP = Online Transaction Processing
ERM = Entity-Relationship-Model
ME/R = Multidimensionales Entity-Relationship-Model
MDBMS = multidimensionale Datenbankmanagementsysteme
ACID = Atomicity, consistency, isolation und durability
ROLAP = relationales OLAP
MOLAP = multidimensionales OLAP
HOLAP = hybrides OLAP
WOEID = Where on Earth Identifier
SLT = SAP Landscape Transformation
AWS = Amazon Webservice